UCRL-LR-136903

A Reduced Grid Method for a Parallel Global Ocean General Circulation Model

Michael Everett Wickett

PhD Thesis

U.S. Department of Energy

Lawrence Livermore National Laboratory

December 1, 1999

DISCLAIMER

Work performed under the auspices of the U. S. Department of Energy by the University of California Lawrence Livermore National Laboratory under Contract W-7405-Eng-48.

UCRL-LR-136903
Distribution Category UC-702

A Reduced Grid Method for a Parallel Global Ocean General Circulation Model

Michael Everett Wickett

Manuscript date: December 1, 1999

LAWRENCE LIVERMORE NATIONAL LABORATORY
University of California • Livermore, California • 94551

A Reduced Grid Method for a Parallel Global Ocean General Circulation Model

by

Michael Everett Wickett

B.S. (California State University at Sacramento) 1994
M.S. (University of California at Davis) 1996

A dissertation submitted in partial satisfaction of the requirements for the degree
of
Doctor of Philosophy

in

Engineering—Applied Science

in the

Office of Graduate Studies
of the
University of California
Davis

Approved:

Professor Garry H. Rodrigue, Chair
Professor Emeritus Richard M. Christensen
Doctor Philip B. Duffy

Committee in Charge

1999

Michael Everett Wickett

December 1999

Engineering—Applied Science

A Reduced Grid Method for a Parallel Global Ocean General Circulation Model

Abstract

A limitation of many explicit finite-difference global climate models is the timestep restriction caused by the decrease in cell size associated with the convergence of meridians near the poles. A computational grid in which the number of cells in the longitudinal direction is reduced toward high-latitudes, keeping the longitudinal width of the resulting cells as uniform as possible and increasing the allowable timestep, is applied to a three-dimensional primitive equation ocean-climate model. This "reduced" grid consists of subgrids which interact at interfaces along their northern and southern boundaries, where the resolution changes by a factor of three. Algorithms are developed to extend the finite difference techniques to this interface, focusing on the conservation required to perform long time integrations, while preserving the staggered spatial arrangement of variables and the numerics used on subgrids. The reduced grid eliminates the common alternative of filtering high-frequency modes from the solution at high-latitudes to allow a larger timestep and reduces execution time per model step by roughly 20 percent. The reduced grid model is implemented for parallel computer architectures with two-dimensional domain decomposition and message passing, with speedup results comparable to those of the original model. Both idealized and realistic model runs are presented to show the effect of the interface numerics on the model solution. First, a rectangular, mid-latitude, flat-bottomed basin with vertical walls at the boundaries is driven only by surface wind stress to compare

three resolutions of the standard grid to reduced grid cases which use various interface conditions. Next, a similar basin with wind stress, heat, and fresh water forcing is used to compare the results of a reduced grid with those of a standard grid result while exercising the full set of model equations. Finally, global model runs, with topography, forcing, and physical parameters similar to those used for ocean-climate studies, are advanced to a near equilibrium state for both the reduced grid and the standard grid. Differences between the two are presented for typical fields of interest, and very little degradation of the solution due to the reduced grid is observed.

For Robin

Acknowledgements

Thanks to my research advisor, Phil Duffy, for the freedom and trust to pursue the path of my choosing and for the support he has given along the way.

Thanks to my academic advisor, Garry Rodrigue, for his time, instruction, and honest advice over the years.

And thanks to Richard Christensen for his suggestions and his willingness to take the time and effort to be a part of this process.

I am grateful to all those in the department of Applied Science and at LLNL who have worked hard to provide the program which has enabled this work.

I would also like to thank Ken Caldeira and Pete Eltgroth for their support in the transition beyond the student phase.

Thanks to the other members of my qualifying examination committee: Ann Orel, Curt Covey, Su-Tzai Soong, and Cornelis van Dam.

My family deserves a great deal of credit for my getting to this point. You all mean very much to me, and your support is the reason I have been successful.

Contents

List of Figures **ix**

List of Tables **xiii**

1 Introduction **1**
1.1 Overview . . . 3
1.2 Review of Previous Work . . . 6
1.2.1 Grids on the Sphere . . . 7
1.2.2 Nested Grids . . . 8
1.2.3 Comparison . . . 10

2 The Ocean Model **11**
2.1 The Primitive Equations . . . 12
2.1.1 Equations of Motion in a Rotating Coordinate Frame . . . 12
2.1.2 Equations of Motion in Spherical Coordinates . . . 14
2.1.3 Approximations for Ocean Modeling . . . 15
2.1.4 Scaling of the Equations of Motion . . . 17
2.1.5 The Hydrostatic Approximation . . . 18
2.1.6 Final Form of Ocean Primitive Equations . . . 19
2.1.7 Other Forms . . . 20
2.2 Solving the Continuous Equations . . . 21
2.2.1 Boundary Conditions . . . 21
2.2.2 Solving the Rigid-Lid System – Elimination of Pressure . . . 22
2.2.3 Solving the Free-Surface System – Explicit Method . . . 25
2.3 Discretizing the Equations . . . 27
2.3.1 Spatial Arrangement of Variables . . . 28
2.3.2 Specification of Topography . . . 31
2.3.3 Discrete Spatial Operators . . . 32
2.3.4 Flux Form . . . 34
2.3.5 Tracer Transport Equations . . . 35
2.3.6 Baroclinic Momentum Equations . . . 38
2.3.7 Barotropic Momentum and Surface Height Equations . . . 42
2.3.8 Surface Height Filter . . . 44
2.3.9 Temporal Discretization . . . 48
2.4 Parallel Implementation . . . 51

3 The Reduced Grid **54**
3.1 Basic Properties . . . 54
3.1.1 Choosing the Refinement Ratio . . . 55
3.1.2 Grid Definition . . . 56
3.1.3 Interface Location and the Staggered Grid . . . 59
3.1.4 Longitudinal Indexing and the Staggered Grid . . . 61
3.2 Implementation Strategy . . . 65
3.2.1 Dummy Interface Rows . . . 66
3.2.2 Loop Modification . . . 68
3.2.3 An Alternate Strategy . . . 73
3.3 Interface Operations . . . 74
3.3.1 Point-to-Point Fill . . . 75
3.3.2 Point-to-Point Copy . . . 76
3.3.3 Average . . . 77
3.3.4 Interpolation . . . 78
3.3.5 Reduction . . . 81
3.3.6 Expansion . . . 81
3.4 Modifications to the Discrete Equations . . . 82
3.4.1 Initialization . . . 82
3.4.2 Tracer . . . 84
3.4.3 Baroclinic Momentum . . . 90
3.4.4 Barotropic Momentum and Surface Height . . . 98
3.5 Parallel Implementation . . . 103

4 Limited Basin Simulations **107**
4.1 Wind-Driven, Flat Bottomed Basin . . . 109
4.1.1 Kinetic Energy . . . 112
4.1.2 Surface Height . . . 116
4.1.3 Barotropic Velocity . . . 125
4.1.4 Meridional Overturning . . . 134
4.2 Another Wind-Driven, Flat Bottomed Basin . . . 137
4.2.1 Barotropic Velocity . . . 138
4.2.2 Meridional Overturning . . . 146
4.3 Wind and Thermohaline Driven, Flat Bottomed Basin . . . 149
4.3.1 Barotropic Velocity . . . 151
4.3.2 Surface Height . . . 155
4.3.3 Meridional Overturning . . . 159
4.3.4 Temperature and Salinity . . . 162
4.4 Summary of Limited Basin Runs . . . 170

5 Global Simulations **172**
5.1 Model Configuration and Forcing . . . 173
5.2 Temperature and Salinity . . . 183
5.3 Surface Height . . . 195
5.4 Circulation . . . 195
5.5 Parallel Timing Results . . . 203

6 Conclusions **213**
6.1 Summary . . . 213
6.2 Future Work . . . 215

A Numerical Reference **217**
A.1 Conservation . . . 217
A.2 Accuracy . . . 220
A.2.1 Leapfrog Time Differencing . . . 220
A.2.2 Euler Backward Time Differencing . . . 221
A.2.3 Reduced Grid Interface . . . 222
A.3 Stability . . . 224
A.3.1 Leapfrog Method for Advection . . . 225
A.3.2 Euler Backward Method for Advection . . . 227
A.3.3 Leapfrog Method for Diffusion . . . 228
A.3.4 Leapfrog Method for Advection-Diffusion . . . 230
A.3.5 Shallow-Water Equations . . . 232
A.4 Interpolation . . . 234
A.4.1 Formulations for Polynomial Interpolation . . . 234
A.4.2 Accuracy of Polynomial Interpolation . . . 237
A.4.3 Cubic Spline Formulation . . . 239
A.4.4 Cubic Spline Accuracy . . . 242
A.5 Filtering . . . 243

Bibliography **246**

List of Figures

1.1 A global reduced grid 4

2.1 The arrangement of variables in the horizontal 28
2.2 The arrangement of variables in the vertical 30
2.3 An example of bottom topography 31
2.4 Location of some "flux" quantities 33
2.5 An example of a "checkerboard" solution 45
2.6 An example of regular domain decomposition 52

3.1 Even grid stagger 55
3.2 Odd grid stagger 56
3.3 Reduced grid properties for $\Delta x_{\rm ref} = (2/3)\Delta x_{\rm eq}$ 58
3.4 Reduced grid properties for $\Delta x_{\rm ref} = \Delta x_{\rm eq}$ 59
3.5 Relative grid interface locations — type-1 60
3.6 Relative grid interface locations — type-2 60
3.7 Grid overview 62
3.8 Tracer grid (sub)domain 63
3.9 Velocity grid (sub)domain 64
3.10 An interface with no dummy row 66
3.11 An Interface with a dummy row 67
3.12 Type-1 north and south facing interface latitudinal indexing 69
3.13 Type-2 north and south facing interface latitudinal indexing 70
3.14 An example two-dimensional, reduced grid array 71
3.15 Point-to-point fill 75
3.16 Point-to-point copy 76
3.17 Averaging 77
3.18 Interpolation 78
3.19 Interpolation near topography 80
3.20 Setting of velocity grid topography at a type-1 interface 83
3.21 Setting of velocity grid topography at a type-2 interface 84
3.22 Calculation of tracer advective velocities at a type-1 interface 86
3.23 Calculation of tracer advective velocities at a type-2 interface 86
3.24 Calculation of tracer advective fluxes at a type-1 interface 88
3.25 Calculation of tracer advective fluxes at a type-2 interface 88
3.26 Calculation of tracer diffusive fluxes at a type-1 interface 89

3.27 Calculation of tracer diffusive fluxes at a type-2 interface 90
3.28 Calculation of velocity grid advective velocities at a type-1 interface 92
3.29 Calculation of velocity grid advective velocities at a type-2 interface 92
3.30 Calculation of velocity grid bottom advective velocities at a type-1 interface 93
3.31 Calculation of velocity grid bottom advective velocities at a type-2 interface 93
3.32 Calculation of longitudinal pressure gradient at a type-1 interface 96
3.33 Calculation of latitudinal pressure gradient at a type-1 interface 97
3.34 Calculation of longitudinal pressure gradient at a type-2 interface 97
3.35 Calculation of latitudinal pressure gradient at a type-2 interface 98
3.36 Calculation of zonal barotropic velocity gradient at a type-1 interface 101
3.37 Calculation of meridional barotropic velocity gradient at a type-1 interface . 101
3.38 Calculation of zonal barotropic velocity gradient at a type-2 interface 102
3.39 Calculation of meridional barotropic velocity gradient at a type-2 interface . 102
3.40 An example of reduced grid domain decomposition 104

4.1 The standard grid for the midlatitude basin 108
4.2 Zonal, linear wind stress 110
4.3 The reduced grid for the midlatitude basin 111
4.4 Linear-wind-forced basin kinetic energy by grid 114
4.5 Linear-wind-forced basin kinetic energy for a type-1 reduced grid interface . 115
4.6 Linear-wind-forced basin kinetic energy for a type-2 reduced grid interface . 115
4.7 Surface height of the linear-wind-forced, fine resolution basin. 119
4.8 Surface height of the linear-wind-forced, standard resolution basin. 120
4.9 "Error" in surface height of the linear-wind-forced, standard basin 120
4.10 Surface height of the linear-wind-forced, coarse resolution basin. 121
4.11 "Error" in surface height of the linear-wind-forced, coarse basin 121
4.12 Surface height of the linear-wind-forced, reduced grid basin 122
4.13 "Error" in surface height of the linear-wind-forced, reduced grid basin . . . 122
4.14 r.m.s. "error" in surface height of the linear-wind-forced basin by grid . . . 123
4.15 r.m.s. "error" in surface height of the linear-wind-forced basin by type-1 interface interpolation type 124
4.16 r.m.s. "error" in surface height of the linear-wind-forced basin by type-2 interface interpolation type 124
4.17 Barotropic velocity of the linear-wind-forced, standard resolution basin. . . 127
4.18 Barotropic velocity of the linear-wind-forced, coarse resolution basin. 127
4.19 Barotropic velocity of the linear-wind-forced, reduced grid basin. 128
4.20 Interface meridional velocity of the linear-wind-forced basin by grid 129
4.21 Interface northward transport of the linear-wind-forced basin by grid 129
4.22 Type-1 interface meridional velocity of the linear-wind-forced basin by interpolation type 130
4.23 Type-2 interface meridional velocity of the linear-wind-forced basin by interpolation type 130
4.24 Type-1 interface northward transport of the linear-wind-forced basin by interpolation type 131
4.25 Type-2 interface northward transport of the linear-wind-forced basin by interpolation type 131
4.26 Interface zonal velocity of the linear-wind-forced basin by grid 132

4.27 Type-1 interface zonal velocity of the linear-wind-forced basin by interpolation type . . . 133
4.28 Type-2 interface zonal velocity of the linear-wind-forced basin by interpolation type . . . 133
4.29 Meridional overturning of the linear-wind-forced, standard resolution basin. Positive values indicate clockwise circulation. . . . 135
4.30 "Error" in overturning of the linear-wind-forced, coarse basin . . . 136
4.31 "Error" in overturning of the linear-wind-forced, reduced grid basin . . . 136
4.32 Zonally constant wind stress . . . 137
4.33 Barotropic velocity of the "real"-wind-forced, standard resolution basin. . . 140
4.34 Interface meridional velocity of the "real"-wind-forced basin by resolution . 141
4.35 Interface northward transport of the "real"-wind-forced basin by resolution 141
4.36 Type-1 interface meridional velocity of the "real"-wind-forced basin by interpolation type . . . 142
4.37 Type-2 interface meridional velocity of the "real"-wind-forced basin by interpolation type . . . 142
4.38 Type-1 interface northward transport of the "real"-wind-forced basin by interpolation type . . . 143
4.39 Type-2 interface northward transport of the "real"-wind-forced basin by interpolation type . . . 143
4.40 Interface zonal velocity of the "real"-wind-forced basin by resolution . . . 144
4.41 Type-1 interface zonal velocity of the "real"-wind-forced basin by interpolation type . . . 145
4.42 Type-2 interface zonal velocity of the "real"-wind-forced basin by interpolation type . . . 145
4.43 Meridional overturning of the "real"-wind-forced, standard resolution basin. 147
4.44 "Error" in overturning of the "real"-wind-forced, coarse resolution basin . . 148
4.45 "Error" in overturning of the "real"-wind-forced, reduced grid basin . . . 148
4.46 Surface temperature and salinity for the fully-forced basin . . . 150
4.47 Barotropic velocity of the fully-forced, standard resolution basin. . . . 152
4.48 Barotropic velocity of the fully-forced, coarse resolution basin. . . . 152
4.49 Barotropic velocity of the fully-forced, reduced grid basin. . . . 153
4.50 Interface meridional velocity of the fully-forced basin . . . 153
4.51 Interface northward transport of the fully-forced basin . . . 154
4.52 Interface zonal velocity of the fully-forced basin . . . 154
4.53 Surface height of the fully-forced, standard resolution basin. . . . 156
4.54 "Error" in surface height of the fully-forced, coarse resolution basin . . . 157
4.55 "Error" in surface height of the fully-forced, reduced grid basin . . . 157
4.56 r.m.s. "error" in surface height of the fully-forced basin . . . 158
4.57 Meridional overturning of the fully-forced, standard resolution basin . . . 160
4.58 "Error" in overturning of the fully-forced, coarse resolution basin . . . 161
4.59 "Error" in overturning of the fully-forced, reduced grid basin . . . 161
4.60 Temperature at level 2 of the fully-forced, standard resolution basin . . . 164
4.61 "Error" in temperature at level 2 of the fully-forced, coarse basin . . . 165
4.62 "Error" in temperature at level 2 of the fully-forced, reduced grid basin . . 165
4.63 Salinity at level 2 of the fully-forced, standard resolution basin . . . 166
4.64 "Error" in salinity at level 2 of the fully-forced, coarse basin . . . 167

4.65 "Error" in salinity at level 2 of the fully-forced, reduced grid basin 167
4.66 r.m.s. "error" in surface temperature of the fully-forced basin 168
4.67 r.m.s. "error" in temperature at bottom layer of the fully-forced basin . . . 168
4.68 r.m.s. "error" in surface salinity of the fully-forced basin 169
4.69 r.m.s. "error" in salinity at bottom layer of the fully-forced basin 169

5.1 Global topography 179
5.2 Annual mean wind stress from Hellerman and Rosenstein 179
5.3 Zonal mean initial temperature from Levitus 180
5.4 Zonal mean initial salinity from Levitus 181
5.5 Basin definition 182
5.6 Global average temperature versus time. 183
5.7 Global average salinity versus time. 184
5.8 Global zonal mean temperature comparison 187
5.9 Atlantic zonal mean temperature comparison 188
5.10 Pacific zonal mean temperature comparison 189
5.11 Global zonal mean salinity comparison 190
5.12 Atlantic zonal mean salinity comparison 191
5.13 Pacific zonal mean salinity comparison 192
5.14 Surface temperature comparison 193
5.15 Surface salinity comparison 194
5.16 Global surface height 196
5.17 Global meridional overturning 198
5.18 Atlantic meridional overturning 199
5.19 Pacific meridional overturning 200
5.20 Depth averaged zonal velocity at 60 degrees south 201
5.21 Depth averaged meridional velocity at 60 degrees south 201
5.22 Surface zonal velocity at 60 degrees south 202
5.23 Surface meridional velocity at 60 degrees south 202
5.24 Comparison of Standard and Reduced grid parallel execution speed. 205
5.25 Percentage reduced grid speedup 206
5.26 Comparison of Standard and Reduced grid parallel speedup. 207
5.27 Percentage of time used for communication: Standard and Reduced 208
5.28 Standard grid parallel speedup 209
5.29 Reduced grid parallel speedup 209
5.30 Comparison of standard and barotropic filtered grid parallel speedup. . . . 211
5.31 Percentage of time used for communication: with and without barotropic filtering 212

A.1 An Interface 223
A.2 Staggered grid for shallow-water equations 232
A.3 Interpolation geometry 234
A.4 Interpolation points 237

List of Tables

4.1 Basin depth levels 107
4.2 Parameter values for the linear, wind-forced basin. 109
4.3 Resolution of the four basin cases 110
4.4 Steady-state kinetic energies for the linear-wind-forced basin 113
4.5 Steady-state kinetic energies for the fully-forced basin 151

5.1 Depth levels for the global run 174
5.2 Mixing parameters for the global runs. 175
5.3 Flows in the global run 195
5.4 Standard grid timing results 204
5.5 Reduced grid timing results 204
5.6 Standard grid, barotropic filtered timing results 211

Chapter 1

Introduction

Our planet is largely covered by a shallow layer of water that is gathered into large, connected basins. The sun, warming both the atmosphere and the water itself, provides energy which drives circulation of the waters through wind stresses, heating, and the addition and removal of water. The resulting general circulation is a basin scale flow pattern with time scales of hundreds to thousands of years. There has long been interest in understanding the workings of the world ocean. From transportation and military strategy to fisheries and climatic influences, the ocean's currents and fluid properties are of vital importance to our knowledge of the planet and the effects our activities have on it [71].

The human population is becoming increasingly aware of its ability to impact the world around us. Extinction of species from pollution, large changes in ecosystems from development, and impact on climate by fossil fuel emissions are some of the problems humanity is learning to understand and limit. Man's impact on climate is currently driving the greatest interest in the global ocean circulation [34]. The oceans act as a storage reservoir for both water and heat, as well as other materials of climatic importance, notably

carbon dioxide. This reservoir interacts with the atmosphere in complex ways which sometimes reduces and sometimes strengthens changes to the system. The effects of increased carbon dioxide in the atmosphere from fossil-fuel burning and the resulting changes to atmospheric processes cannot be accurately predicted without considering ocean-atmosphere interactions [29].

In centuries past, observations compiled on ships yielded most of the available knowledge about the sea. As better instruments have been developed, greater understanding of the deeper realms of the oceans has been gained. Recently, satellite data acquisition has given increased precision and quantities of some types of ocean data, such as surface height and the extent of sea-ice [53]. Meanwhile, large strides have also been made in the theory of the general circulation. Major features have been explained well by simplified models, and the governing equations have become better understood. Computers, however, have proven invaluable in studying the complex system of equations that describe the oceans, since analytic solutions are not possible for the full equations, nor even for simplified forms of the equations in realistic geometries [31].

Computational power has increased to a level which enables global ocean models and global atmospheric models to be coupled together and integrated for times useful to the basic understanding of human climate influence, i.e. for centuries of simulated time [25, 73]. However, these models are still computationally very expensive, and in order to run these simulations, only features on the order of 100 kilometers can be explicitly resolved. At the same time, other researchers have shown that much finer resolution simulations are possible which allow study of smaller scale features [60]. But for now these high-resolution models are limited to smaller regions and simulation lengths. The amount of computer power required to run numerical models remains a major limitation in the advancement

of climate modeling and will likely remain so for the foreseeable future.

1.1 Overview

This dissertation describes the design and use of a method for reducing the computer time needed to integrate a global ocean general circulation model. The Bryan-Cox type model (see Chapter 2), and most other ocean models in use, solve the equations describing the global ocean by the method of explicit finite differences. In both ocean and atmosphere models, the equations are generally solved in the natural coordinate system of spherical coordinates where a position on the Earth is given by its latitude, longitude, and depth or height from the surface. The combination of this coordinate system and explicit finite difference methods lead to what is termed the "pole problem."

The convergence of meridians toward the poles in spherical coordinates causes two major problems. First, at the pole itself, the coordinate system has a singularity, with direction becoming undefined. This can be handled in a few different ways, including averaging of neighboring velocities or using a mesh with variables arranged such that there is no velocity point at the pole. It is handled quite simply in the model used here by placing a land cell, i.e. a zero velocity point, at the pole. The second problem is due to the dependence of the stability conditions from finite difference numerics on the size of grid cells. As the minimum cell size decreases, so too does the allowable timestep that can be used in the model (see Section A.3). At the highest latitudes in the spherical system, the grid cells have a width which is less than ten percent the size of the equatorial cells; this forces the use of a timestep which is much smaller than that needed in most grid cells.

One method of allowing the use of larger timesteps without instability is the filtering of

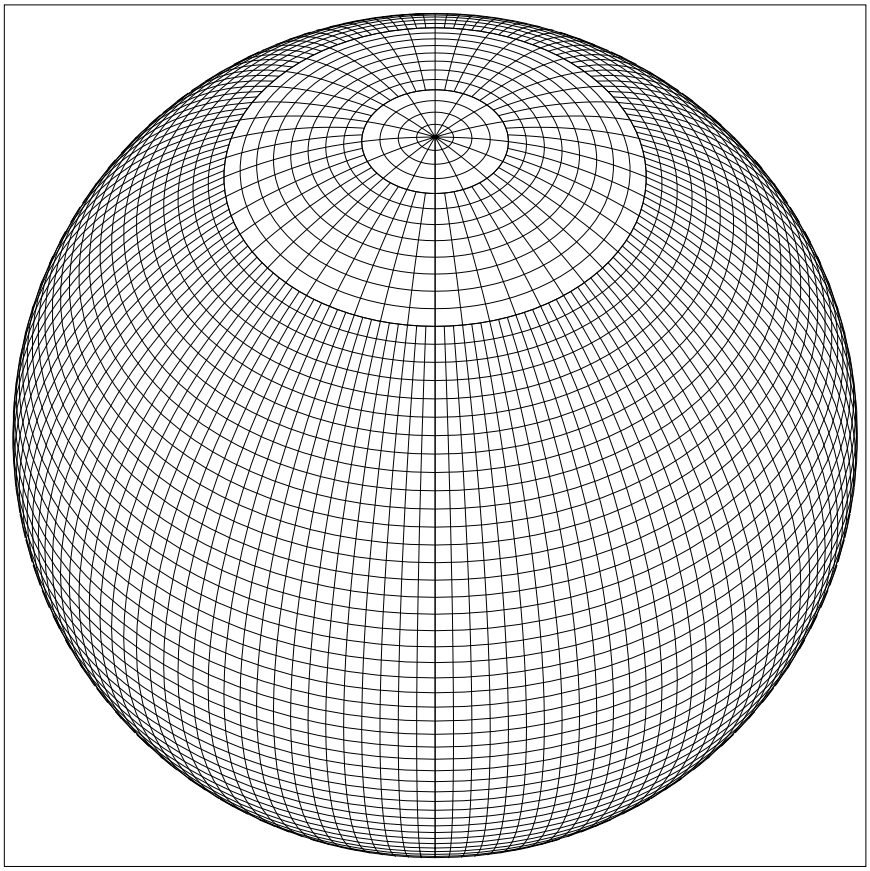

Figure 1.1: A global reduced grid

high frequency components from the model solutions (see Section A.5). In this approach, a timestep which would normally violate the stability criteria at high latitudes is used, and the high-frequency components of the solution which would cause instability are removed at each timestep via filtering. This technique is frequently used in global ocean and atmosphere models and is the standard configuration of the model used as the starting point of this work. Filtering is thus used in the control case of the global runs described in this work in Chapter 5. Disadvantages of filtering include the need for additional computation, possible damage to the model solution, as well as poor scaling and difficult load-balancing on parallel distributed memory computers.

A method which will not only increase the allowable timesteps given by numerical stability but also decrease the number of grid cells used by the model is here termed the "reduced grid" method. On this grid, an example of which is shown in Figure 1.1, the cell size is kept much more uniform by decreasing the number of grid cells in the longitudinal direction as the pole is approached. Because the grid is only modified at a small number of latitudes, the numerics on the majority of the grid are unmodified. Due to the smaller number of grid cells and the lack of filtering, less computation is required. The interactions between regions of differing resolution are similar to those found in models using composite meshes or mesh refinement that can be found in fluid dynamics and engineering [3, 4, 64]. The design, implementation, and testing of a reduced grid method in the Lawrence Livermore National Laboratory global ocean general circulation model is the subject of this work.

Following this overview, previous work of a related nature is summarized. Then Chapter 2 describes the basic ocean model, the solution method and numerics it uses, and its implementation on parallel, distributed memory computer architectures. Chapter 3 then details the approach to implementing the reduced grid in this model, the modifications to the numerical algorithms required, and the modifications it requires on parallel architectures. Model runs are presented in Chapter 4 and Chapter 5, which progress from limited basin runs with minimal forcing to extended simulations of the global model with realistic topography and forcing. Timing comparisons of the base model and the reduced grid model, including parallel speedup are also given in Chapter 5. Then Chapter 6 gives a summary of the results and ideas of what they might lead to in the future.

1.2 Review of Previous Work

Ocean circulation models forced by both surface wind stress and buoyancy fluxes have been in use for over thirty years. In many ways they have followed the development of atmospheric models which are used for weather and climate prediction [44]. Models which have a global or nearly global domain have been in use for nearly twenty-five years [15, 68]. As computing power has grown, so has the ambition of modelers. Integration times of simulations have grown, while spatial resolution has increased. The ever-present limitation of computer resources has led to many novel methods for faster and more accurate simulations [59].

There are other numerical techniques used in ocean and atmosphere modeling which do not suffer from the timestep restriction caused by the spherical coordinate system. These include the use of other coordinate systems, described in the following section, as well as spectral methods, implicit methods, and finite element methods. While atmospheric models routinely make use of the spectral method (see [8] and [1]), no major global ocean models use this method. However, Iskandarani et al. [35] combined the spectral method with finite elements in a model using the oceanic shallow-water equations. They also tested both explicit and implicit time integration, finding that the implicit method takes more computation time. In general it is thought that implicit methods take too much computation time for the large, nonlinear systems used in ocean modeling.

The two distinct categories of previous work which relate directly to the present work are discussed in detail below. One deals with the reduction or elimination of the "pole problem" associated with meshes in spherical coordinates. Approaches which have been tested include modifications of the spherical polar system and projections of other co-

ordinate systems onto the sphere. The other category is concerned with regular meshes nested within one another to achieve flexibility in the model resolution. Many nested models have been created for use in regional modeling with the purpose of moving the domain boundaries away from the region of interest without greatly increasing computational cost. In some models these nested grids are allowed to move in order to track features of the solution.

1.2.1 Grids on the Sphere

As far back as 1959, Kuo and Nordo [38] integrated a simple atmosphere model over one hemisphere with a grid using a Mercator projection and doubling of the grid spacing at three latitudes moving northward. No details of the specific conditions at the interfaces were given, and a computational instability prevented integration beyond five days. The cause of the instability was speculated to be the grid coarsening, but no analysis was given. A few years later, Gates and Riegel [27, 28] integrated barotropic flow equations over the northern hemisphere on a grid whose longitudinal mesh spacing varied with latitude. The jumps in mesh spacing were not always by integer factors, and they avoided the use of interpolation by using a simplified procedure for derivatives near the interface.

Kurihara [39], assuming that a homogeneous density of grid points on the globe was a desirable property for a barotropic atmosphere model, created a grid system in which the number of cells varied between each latitude row. This system was integrated for sixteen days with conservation of model quantities. Following this, Holloway, Spelman, and Manabe [33] addressed the problems they saw with the Kurihara grid. Suggesting possible causes for the problems, they comment on the difference in phase speeds of waves at each latitude. In response, they implemented a latitude-longitude grid with filtering

of all the model variables at high-latitudes to overcome the timestep restriction of the convergence of meridians. The resulting model ran more slowly than the Kurihara model, but their opinion was that the programming was easier than other global grid systems. This is the method which came to be widely used in many finite-difference ocean and atmosphere models.

Browning, Hack, and Swarztrauber [9] compared two spectral transform methods to a method which used finite-differences on an overlapping stereographic coordinate system. Then Rancic, Purser, and Mesinger [56] presented a shallow-water model using a composite grid which is the mapping of a cube onto the surface of the sphere. Swarztrauber, Williamson, and Drake [66] solved the shallow-water equations by transforming the two-dimensional, spherical coordinate system to a three-dimensional, Cartesian coordinate system in which the computation is limited to the surface of the sphere by projecting the equations, gradients, and solution onto the surface. These methods of coordinate transformation have been limited to simpler models, and no global solutions of the primitive equations for ocean modeling have been performed with these methods in the literature.

Rasch [57] presented a new discretization of the two-dimensional transport equation and applied it on a reduced grid. His reduced grid had a coarsening ratio of two and no staggering of variables. Tests involving two-dimensional transport over the pole with a constant wind field as well as a compact field undergoing solid-body rotation were presented. No follow-up work has been published using the reduced grid in an application.

1.2.2 Nested Grids

Sobel [62] presented a variety of work with a nested atmospheric model, both in two and three dimensions. A staggered grid was used, and model quantities were conserved.

Multiple methods of smoothing were also used. Kurihara, Tripoli, and Bender [40] described a movable, nested-mesh technique for primitive equations on an unstaggered mesh which conserves mass, momentum, and internal energy. They applied this to a one-dimensional shallow-water equation model, allowing the nested mesh to follow a perturbation of the geopotential field. Then Zhang, Chang, Seaman, Warner, and Fritsch [76] described a nested atmospheric mesoscale model which uses a staggered mesh with a refinement ratio of three. At interfaces, solution tendencies were interpolated and damping was applied, but mass and energy were not conserved. A procedure for dealing with terrain for the differing grid resolutions was presented.

Spall and Holland [63] introduced a nested primitive equation model for oceanic applications and applied it to two test cases. Their model did not conserve quantities at the interfaces of their nested grids, as they considered the smoothness of the solution passing between grids to be more important for their application. They noted, however, that for some applications, namely those requiring longer time integrations, conservation is likely to be a more important issue. Similarly, Fox and Maskell [26] presented another nested ocean model using a staggered grid with both horizontal and vertical refinement of the mesh. They too did not conserve quantities at the grid interfaces, focusing instead on smoothness of the solution. A damping term was added at the interface to further provide for smooth solutions in the vicinity of the interfaces. Ginis, Richardson, and Rothstein [30] presented an ocean model with multiply-nested grids. Their model uses an unstaggered grid, but does conserve mass, momentum, heat, and salinity at the interfaces. Similar models are described in [47] and [41].

Finally, Blayo and Debreu [7] applied adaptive mesh refinement (AMR) to the field of ocean modeling. The mesh refinement part of AMR is very similar to the present

reduced grid method. Test cases using the shallow-water and quasi-geostrophic equations were presented. They posed some interesting questions as to the possibility of using AMR for long-time integrations of primitive equation ocean models, including dealing with topography at different refinement levels and choosing the best criteria for refinement. They are planning to implement this technique in a limited basin, primitive equation model soon. See [3], [4], [5], [6], and [43] for further information on AMR.

1.2.3 Comparison

All of the works described above have features similar to those of the present paper. In the papers of Section 1.2.1, barotropic, primitive equation, or other geophysical models have been developed using varying grid resolution or alternative grids to alleviate problems with solution on the sphere. Those in Section 1.2.2 develop models containing grids with varying resolution and interfaces between those grids. The present work is distinguished by the combination of several factors: (1) use of full three-dimensional primitive equations; (2) the "staggered mesh" arrangement of variables; (3) applicability to a global domain and long time integrations; and (4) implementation on parallel computer architectures. While some of these features are shared with the work described above, no previous work has implemented a reduced grid in a parallel ocean model and successfully integrated this model to equilibrium for a global domain.

Chapter 2

The Ocean Model

Since the reduced grid method has been implemented in a mature ocean climate model, the standard model will be described in this chapter. First the primitive equations of the model are derived from the Navier-Stokes equations. Then the general solution procedure to the model equations is presented. Next the numerical discretization is given which implements this solution procedure. The method of implementation on parallel computer architectures is then explained. This information will provide a reference for later chapters and a basis with which to compare the reduced grid method.

The model used as the basis for this work is the Lawrence Livermore National Laboratory (LLNL) ocean general circulation model. This model is based on the Modular Ocean Model (MOM) from the Geophysical Fluid Dynamics Laboratory (GFDL). This type of ocean model is often referred to as a Bryan-Cox ocean model (see [11] and [15]) and is generally accepted as the most widely used type of ocean general circulation model. Details of the LLNL ocean model and results from its use can be found in [13], [14], [19], [20], [21], [22], [23], and [24]. A significant feature of the climate model is its ability to operate efficiently on many high-performance computer architectures with little to no modification.

See [25], [45], and [74] for performance and design issues.

2.1 The Primitive Equations

The primitive equations for the modeling of the general ocean circulation will be derived here from the basic equations of fluid dynamics. For a general discussion of these basic equations, refer to [51] or [75]. The first equation is the continuity equation, i.e. the equation for the conservation of mass,

$$\frac{\partial \rho}{\partial t} = -\nabla \cdot \rho \mathbf{u}. \tag{2.1}$$

The Navier-Stokes equations describe the conservation of momentum and are given for a cartesian coordinate system by

$$\frac{\partial u_i}{\partial t} + u_j \frac{\partial u_i}{\partial x_j} = -\frac{1}{\rho}\frac{\partial p}{\partial x_i} - \frac{\partial \Phi_N}{\partial x_i} + F_i, \tag{2.2}$$

where F is the frictional force and Φ_N is the gravitational potential. The conservation of heat, salt, and chemical tracers are described by the more simple transport equation,

$$\frac{\partial T}{\partial t} + u_j \frac{\partial T}{\partial x_j} = D, \tag{2.3}$$

where T is the tracer quantity, and D is a diffusion term. The density of the fluid is related to the temperature, salinity, and pressure by an equation of state, i.e. $\rho = \rho(T, S, p)$. See [29] for a discussion of the equation of state. The model uses the equation of state set by the Joint Panel on Oceanographics Tables and Standards (UNESCO, 1981).

2.1.1 Equations of Motion in a Rotating Coordinate Frame

Since the Earth is rotating, it is convenient to convert the equations to a coordinate system that is also rotating. Let Ω be the rotation vector of the Earth. Then for any

vector, $\mathbf{v}$, the transformation of the time rate of change of the vector from the rotating to the inertial frame is given by

$$\frac{d\mathbf{v}_a}{dt} = \frac{d\mathbf{v}}{dt} + \Omega \times \mathbf{v}, \tag{2.4}$$

where the subscript a denotes a vector in the inertial frame, while no subscript denotes a vector in the rotating frame. Let $\mathbf{r}$ be the position vector of an arbitrary fluid element. According to the transformation

$$\frac{d\mathbf{r}_a}{dt} = \frac{d\mathbf{r}}{dt} + \Omega \times \mathbf{r}, \tag{2.5}$$

so that the velocity seen in the nonrotating frame, $d\mathbf{r}_a/dt = \mathbf{u}_a$, is equal to the velocity seen in the rotating frame, $d\mathbf{r}/dt = \mathbf{u}$, plus the velocity of the fluid element due to the rotation. Then (2.5) can be written as

$$\mathbf{u}_a = \mathbf{u} + \Omega \times \mathbf{r}. \tag{2.6}$$

Applying (2.4) to $\mathbf{u}_a$ gives

$$\frac{d\mathbf{u}_a}{dt} = \frac{d}{dt}\left(\mathbf{u} + \Omega \times \mathbf{r}\right) + \Omega \times \left(\mathbf{u} + \Omega \times \mathbf{r}\right), \tag{2.7}$$

or,

$$\frac{d\mathbf{u}_a}{dt} = \frac{d\mathbf{u}}{dt} + 2\Omega \times \mathbf{u} + \Omega \times (\Omega \times \mathbf{r}), \tag{2.8}$$

where Ω is assumed to be constant. Thus, an observer in the rotating frame of reference observes two additional acceleration terms. They are the Coriolis acceleration, $2\Omega \times \mathbf{u}$, and the centripetal acceleration, $\Omega \times (\Omega \times \mathbf{r})$. The centripetal acceleration term can be written as

$$\Omega \times (\Omega \times \mathbf{r}) = -\nabla \Phi_c \tag{2.9}$$

$$\text{where} \quad \Phi_c = \frac{|\Omega \times \mathbf{r}|^2}{2}.$$

Putting the results (2.8) and (2.9) into the Navier-Stokes equations (2.2) gives

$$\frac{\partial u_i}{\partial t} + u_j \frac{\partial u_i}{\partial x_j} = -\frac{1}{\rho}\frac{\partial p}{\partial x_i} - \frac{\partial \Phi_N}{\partial x_i} + F_i - 2(\Omega \times u)_i + \frac{\partial \Phi_c}{\partial x_i}. \tag{2.10}$$

Now the gravitional and centripetal potentials can be combined. Let

$$\Phi = \Phi_N - \Phi_c \tag{2.11}$$

and

$$\mathbf{g} = -\nabla \Phi, \tag{2.12}$$

so that (2.10) becomes

$$\frac{\partial u_i}{\partial t} + u_j \frac{\partial u_i}{\partial x_j} = -\frac{1}{\rho}\frac{\partial p}{\partial x_i} + g_i + F_i - 2(\Omega \times u)_i. \tag{2.13}$$

2.1.2 Equations of Motion in Spherical Coordinates

When describing a position on the Earth, it is customary to use a modified form of the standard spherical polar coordinates with position defined as (λ, ϕ, z), with

$\lambda =$ longitude increasing eastward

$\phi =$ latitude increasing northward with 0 at equator

$z =$ height increasing upwards with 0 at surface.

Then the velocities are given by

$$u = R\cos\phi\,\frac{\partial \lambda}{\partial t} \tag{2.14}$$

$$v = R\,\frac{\partial \phi}{\partial t} \tag{2.15}$$

$$w = \frac{\partial z}{\partial t}, \tag{2.16}$$

where R is the earth radius. With these definitions, the vector operators become

$$\nabla q = \frac{1}{R\cos\phi}\frac{\partial q}{\partial \lambda}\hat{\lambda} + \frac{1}{R}\frac{\partial q}{\partial \phi}\hat{\phi} + \frac{\partial q}{\partial z}\hat{z} \tag{2.17}$$

$$\nabla \cdot \mathbf{v} = \frac{1}{R\cos\phi}\frac{\partial v_\lambda}{\partial \lambda} + \frac{1}{R\cos\phi}\frac{\partial (v_\phi \cos\phi)}{\partial \phi} + \frac{\partial v_z}{\partial z} + \frac{2v_z}{R} \tag{2.18}$$

$$\nabla^2 q = \frac{1}{R^2\cos^2\phi}\frac{\partial^2 q}{\partial \lambda^2} + \frac{1}{R^2\cos\phi}\frac{\partial}{\partial \phi}\left(\cos\phi\frac{\partial q}{\partial \phi}\right) + \frac{\partial^2 q}{\partial z^2} + \frac{2}{R}\frac{\partial q}{\partial z}. \tag{2.19}$$

Define the Coriolis parameters

$$f = 2\Omega\sin\phi \tag{2.20}$$

$$f' = 2\Omega\cos\phi, \tag{2.21}$$

so that

$$2(\Omega \times \mathbf{u})_\lambda = f'w - fv \tag{2.22}$$

$$2(\Omega \times \mathbf{u})_\phi = fu \tag{2.23}$$

$$2(\Omega \times \mathbf{u})_z = f'u. \tag{2.24}$$

Then the momentum equations become

$$\frac{Du}{Dt} = \frac{\tan\phi}{R}uv - \frac{uw}{R} + fv - f'w - \frac{1}{\rho R\cos\phi}\frac{\partial p}{\partial \lambda} + g_\lambda + F_\lambda \tag{2.25}$$

$$\frac{Dv}{Dt} = -\frac{\tan\phi}{R}u^2 - \frac{vw}{R} - fu - \frac{1}{\rho R}\frac{\partial p}{\partial \phi} + g_\phi + F_\phi \tag{2.26}$$

$$\frac{Dw}{Dt} = \frac{u^2}{R} + \frac{v^2}{R} - f'u - \frac{1}{\rho}\frac{\partial p}{\partial z} + g_z + F_z, \tag{2.27}$$

where the material derivative, $\frac{D}{Dt}$, is defined as

$$\frac{D}{Dt} = \frac{\partial}{\partial t} + \frac{u}{R\cos\phi}\frac{\partial}{\partial \lambda} + \frac{v}{R}\frac{\partial}{\partial \phi} + w\frac{\partial}{\partial z}. \tag{2.28}$$

2.1.3 Approximations for Ocean Modeling

Now the equations are in a form that is useful for describing motion on a rotating sphere, but they are still too complex for modeling purposes. There are, however, a number of approximations that can be made in order to simplify them.

First, note that the horizontal components of the gravitational acceleration, i.e. g_λ and g_ϕ, are very small compared to the other terms in the horizontal momentum equations, and can therefore be neglected.

The vertical scale of the motion is everywhere much smaller than the radius of the earth, so that the gravitational acceleration may be assumed to be constant with depth.

The density of seawater is very nearly constant throughout the ocean, so an approximation, known as the Boussinesq approximation, can be made in which $\rho = \rho_0 =$ constant in all terms except for the buoyancy term, g_z. In that term, ρ is calculated by an equation of state which relates it to the temperature and salinity. With this approximation, the continuity equation, (2.1), expressed in spherical coordinates becomes

$$\nabla \cdot \mathbf{u} = \frac{\partial w}{\partial z} + \frac{2w}{R} + \frac{1}{R\cos\phi}\frac{\partial (v\cos\phi)}{\partial \phi} + \frac{1}{R\cos\phi}\frac{\partial u}{\partial \lambda} = 0. \tag{2.29}$$

Physically, the approximation $\rho = \rho_0$ means that this equation now expressed conservation of volume, not mass.

The turbulent viscosity hypothesis is used to represent sub-grid scale motions, that is, motions that are too small to be resolved by the grid. Ordinary viscosity is a very much smaller effect and is consequently ignored. Thus the transfer of energy to these smaller scale motions is modeled as viscous terms of the form,

$$F_i = \frac{1}{\rho}\frac{\partial \tau_{ji}}{\partial x_j} = \nabla_h \cdot (A \nabla_h u_i) + \frac{\partial}{\partial z}\left(\kappa \frac{\partial u_i}{\partial z}\right). \tag{2.30}$$

Then using all of the approximations so far, the equations of motion become

$$\frac{Du}{Dt} = \frac{\tan\phi}{R}uv - \frac{uw}{R} + fv - f'w - \frac{1}{\rho_0 R\cos\phi}\frac{\partial p}{\partial \lambda} + F_\lambda \tag{2.31}$$

$$\frac{Dv}{Dt} = -\frac{\tan\phi}{R}u^2 - \frac{vw}{R} - fu - \frac{1}{\rho_0 R}\frac{\partial p}{\partial \phi} + F_\phi \tag{2.32}$$

$$\frac{Dw}{Dt} = \frac{u^2}{R} + \frac{v^2}{R} - f'u - \frac{1}{\rho_o}\frac{\partial p}{\partial z} - \frac{\rho}{\rho_0}g + F_z. \tag{2.33}$$

It will be useful to estimate which terms in the equations are the largest and most influential for the present purposes to determine the possibility of even more simplification. Thus the relative sizes of the remaining terms should be compared. In order to do so, some characteristic scales of the ocean must be known. This will be done in the next section.

2.1.4 Scaling of the Equations of Motion

The following list consists of representative scales of motion from observed values:

$$\begin{aligned}
\text{horizontal length scale} = L &\approx 10^6 \text{ m} \\
\text{depth scale} = H &\approx 4 \times 10^3 \text{ m} \\
\text{horizontal velocity scale} = U &\approx 10^{-1} \text{ m s}^{-1} \\
\text{vertical velocity scale} = W &\approx 10^{-4} \text{ m s}^{-1} \\
\text{horizontal pressure scale} = \Delta p_h &\approx 10^3 \text{ Pa} \\
\text{vertical pressure scale} = \Delta p_v &\approx 4 \times 10^4 \text{ Pa} \\
\text{time scale} = t &\approx L/U \approx 10^7 \text{ s}
\end{aligned}$$

With $f \approx 10^{-4}$ at midlatitudes, and typical values (for modeling purposes) of A and κ being 10^3 to 10^4 m s^{-2} and 2×10^{-3} m s^{-2}, respectively, then approximations for various terms in (2.31) through (2.33) are

$$\begin{aligned}
\left(\frac{Du}{Dt}, \frac{Dv}{Dt}\right) &\sim \frac{U^2}{L} &&\approx 10^{-8} \\
\left(\frac{Dw}{Dt}\right) &\sim \frac{UW}{L} &&\approx 10^{-11} \\
\left(fu, fv, f'u\right) &\sim fU &&\approx 10^{-5} \\
\left(f'w\right) &\sim fW &&\approx 10^{-8}
\end{aligned}$$

$$
\begin{aligned}
\left(\frac{\tan\phi}{R}uv, \frac{\tan\phi}{R}u^2, \frac{u^2}{R}, \frac{v^2}{R}\right) &\sim \frac{U^2}{R} && \approx 10^{-9} \\
\left(\frac{uw}{R}, \frac{vw}{R}\right) &\sim \frac{UW}{R} && \approx 10^{-12} \\
\left(\frac{1}{\rho_0 R\cos\phi}\frac{\partial p}{\partial\lambda}, \frac{1}{\rho_0 R}\frac{\partial p}{\partial\phi}\right) &\sim \frac{\Delta p_h}{L} && \approx 10^{-3} \\
\left(\frac{1}{\rho_o}\frac{\partial p}{\partial z}\right) &\sim \frac{\Delta p_v}{H} && \approx 10 \\
\left(\frac{\rho}{\rho_0}g\right) &\sim g && \approx 10 \\
(\,\nabla_h\cdot(A\nabla_h u), \nabla_h\cdot(A\nabla_h v)\,) &\sim \frac{AU}{L^2} && \approx 10^{-10} \text{ to } 10^{-9} \\
\left(\frac{\partial}{\partial z}\left(\kappa\frac{\partial u}{\partial z}\right), \frac{\partial}{\partial z}\left(\kappa\frac{\partial v}{\partial z}\right)\right) &\sim \frac{\kappa U}{H^2} && \approx 10^{-11} \\
(\,\nabla_h\cdot(A\nabla_h w)\,) &\sim \frac{AW}{L^2} && \approx 10^{-13} \text{ to } 10^{-12} \\
\left(\frac{\partial}{\partial z}\left(\kappa\frac{\partial w}{\partial z}\right)\right) &\sim \frac{\kappa W}{H^2} && \approx 10^{-14},
\end{aligned}
$$

with all in units of m s^{-2}. For modeling, most of the terms in the horizontal equations are retained, with only those metric terms invlolving w being dropped. In the vertical equation, however, the pressure gradient and gravitational terms clearly dominate, by nearly 10 orders of magnitude, the other terms. This leads to what is termed the hydrostatic approximation.

2.1.5 The Hydrostatic Approximation

In the vertical momentum equation (2.33), the estimates above clearly indicate the dominance of two terms. Dropping all others leaves

$$\frac{\partial p}{\partial z} = -\rho g, \tag{2.34}$$

which is known as the hydrostatic equation. This equation gives the pressure field from the density field. Remembering that the density is found by an equation of state of the

form

$$\rho = \rho(T, S, p), \tag{2.35}$$

it can be calculated based on the tracer fields and then (2.34) can be integrated from the surface (where constant atmospheric pressure is assumed) downward to calculate the pressure field.

Now taking the continuity equation, (2.29),

$$\nabla \cdot \mathbf{u} = \frac{\partial w}{\partial z} + \frac{2w}{R} + \frac{1}{R\cos\phi}\frac{\partial(v\cos\phi)}{\partial\phi} + \frac{1}{R\cos\phi}\frac{\partial u}{\partial\lambda} = 0, \tag{2.36}$$

and using the scaling from above, all terms in this equation are found to be of order 10^{-7} m s^{-2}, except for the second on the right-hand side, which is 4 orders of magnitude smaller. Therefore it can be neglected, giving the approximate continuity equation

$$\frac{\partial w}{\partial z} = -\frac{1}{R\cos\phi}\left(\frac{\partial(v\cos\phi)}{\partial\phi} + \frac{\partial u}{\partial\lambda}\right), \tag{2.37}$$

which can be used to calculate the vertical velocity from the horizontal velocities.

2.1.6 Final Form of Ocean Primitive Equations

Writing the results in complete detail yields

$$\frac{\partial u}{\partial t} + ADV(u) - \frac{uv\tan\phi}{R} - fv = -\frac{1}{\rho_0 R\cos\phi}\frac{\partial p}{\partial\lambda} + F_\lambda \tag{2.38}$$

$$\frac{\partial v}{\partial t} + ADV(v) + \frac{u^2\tan\phi}{R} + fu = -\frac{1}{\rho_0 R}\frac{\partial p}{\partial\phi} + F_\phi \tag{2.39}$$

$$\frac{\partial w}{\partial z} = -\frac{1}{R\cos\phi}\left(\frac{\partial u}{\partial\lambda} + \frac{\partial(v\cos\phi)}{\partial\phi}\right) \tag{2.40}$$

$$\frac{\partial T}{\partial t} + ADV(T) = \frac{\partial}{\partial z}\left(\kappa_h\frac{\partial T}{\partial z}\right) + \nabla_h\cdot(A_h\nabla_h T) \tag{2.41}$$

$$\frac{\partial S}{\partial t} + ADV(S) = \frac{\partial}{\partial z}\left(\kappa_h\frac{\partial S}{\partial z}\right) + \nabla_h\cdot(A_h\nabla_h S) \tag{2.42}$$

$$\frac{\partial p}{\partial z} = -\rho g \tag{2.43}$$

$$\rho = \rho(T, S, p), \tag{2.44}$$

where the advective and viscous terms are given by

$$ADV(a) = u\frac{1}{R\cos\phi}\frac{\partial a}{\partial \lambda} + v\frac{1}{R}\frac{\partial a}{\partial \phi} + w\frac{\partial a}{\partial z} \tag{2.45}$$

$$F_\lambda = \nabla_h \cdot (A_m \nabla_h u) + \frac{A_m}{R^2}\left[(1-\tan^2\phi)u - \frac{2v_\lambda \sin\phi}{\cos^2\phi}\right] + \frac{\partial}{\partial z}\left(\kappa_m \frac{\partial u}{\partial z}\right) \tag{2.46}$$

$$F_\phi = \nabla_h \cdot (A_m \nabla_h v) + \frac{A_m}{R^2}\left[(1-\tan^2\phi)v - \frac{2u_\lambda \sin\phi}{\cos^2\phi}\right] + \frac{\partial}{\partial z}\left(\kappa_m \frac{\partial v}{\partial z}\right), \tag{2.47}$$

where A_m and κ_m are the horizontal and vertical viscosity coefficients, and A_h and κ_h are the horizontal and vertical diffusion coefficients, and where ∇_h is the horizontal gradient operator, written in the spherical coordinates as

$$\nabla_h \cdot (a\nabla_h b) = \frac{1}{R^2\cos^2\phi}\frac{\partial}{\partial\lambda}\left(a\frac{\partial b}{\partial\lambda}\right) + \frac{1}{R^2\cos\phi}\frac{\partial}{\partial\phi}\left(a\cos\phi\frac{\partial b}{\partial\phi}\right). \tag{2.48}$$

2.1.7 Other Forms

Other forms of the original equations can be derived by making different assumptions, approximations, or scalings. Two common forms are the shallow-water equations and the quasigeostrophic equations. The former are a set of two-dimensional equations very similar to (2.77), (2.78), and (2.79) given below in the description of the free-surface solution. That is, the shallow-water equations are similar to a two-dimensional subset of the full primitive equations with a free-surface. The quasigeostrophic system describes a near balance between the pressure gradient and Coriolis terms but is limited to less than global applicability due to the failure of this balance near the equator. For the study of global ocean general circulation, these other forms leave out important physical processes and are of limited use. See Pedlosky [51, 52] for more on other geophysical fluid equations and their characteristics.

2.2 Solving the Continuous Equations

This section will describe the general solution method for the primitive equations. First the issue of boundary conditions will be discussed. Following this, two standard approaches for solving the system of primitive equations are described. These will be given without reference to the discretization of the equations. Only the explicit free-surface method of solution will be used in the work described below. However, since the rigid-lid method has seen widespread use in the field, that solution method is outlined as well.

2.2.1 Boundary Conditions

The velocity boundary condition at the ocean bottom is that of no normal flow,

$$\mathbf{u}_n = \mathbf{u} \cdot \nabla H = 0, \tag{2.49}$$

where H is the depth to the bottom. In the model coordinate system this gives

$$w(-H) = -\frac{1}{R\cos\phi} u(-H) \frac{\partial H}{\partial \lambda} - \frac{1}{R} v(-H) \frac{\partial H}{\partial \phi}. \tag{2.50}$$

The bottom tracer boundary conditions are set to zero. The small amount of geothermal heat flux can be ignored. At the surface, the fluxes of heat and fresh water are specified either by another model, such as an ice model or atmospheric model, or from observations. Likewise, the momentum boundary condition at the surface involves specification of wind stresses either from a coupled model or observations.

At the surface, the fluid height can vary, with the height of the fluid governed by

$$w = \frac{\partial \eta}{\partial t} + \frac{u}{R\cos\phi} \frac{\partial \eta}{\partial \lambda} + \frac{v}{R} \frac{\partial \eta}{\partial \phi}, \tag{2.51}$$

η being the surface elevation. Through this condition, surface waves are permitted which have characteristic velocities of $\sqrt{gH}$, where H is the ocean fluid depth. This yields typical velocities of order 100 m s^{-2}, which can be orders of magnitude greater than the maximum velocities seen in the rest of the ocean. While the equations can be discretized and solved directly with this condition, there are consequences on the allowable timestep for explicit finite difference models, which will be addressed later.

Another approach is to eliminate the surface waves by fixing the surface height to be zero. This is known as the "rigid-lid" approximation, effectively increasing the surface wave speed to infinity. This approximation also has further effects on other long-wave dynamics, but these are thought to have little overall effect on the solution. See [37] for further details. Besides these effects, this method also requires one to calculate a two-dimensional streamfunction, which involves the solution of a Poisson-like equation. This will be described in the following section.

2.2.2 Solving the Rigid-Lid System – Elimination of Pressure

Though the free-surface formulation will be used exclusively in this work, the rigid-lid formulation will be outlined for completeness, as it is used in a large number of similar models. To solve the momentum equations, (2.38) and (2.39), the pressure, which is unknown because of the effect of the rigid-lid, must be eliminated. Integrating the continuity equation (2.37) with respect to z from the bottom to the surface gives

$$\begin{aligned} w(0) - w(-H) = &-\frac{1}{R\cos\phi}\left[\frac{\partial}{\partial\lambda}\left(\int_{-H}^{0} u\,dz\right) + \frac{\partial}{\partial\phi}\left(\int_{-H}^{0} v\cos\phi\,dz\right)\right] \\ &+ \frac{1}{R\cos\phi}u(-H)\frac{\partial H}{\partial\lambda} + \frac{1}{R}v(-H)\frac{\partial H}{\partial\phi}. \end{aligned} \tag{2.52}$$

Using $w(0) = 0$ and (2.50), this becomes

$$\frac{\partial}{\partial \lambda}\left(\int_{-H}^{0} u\, dz\right) + \frac{\partial}{\partial \phi}\left(\int_{-H}^{0} v \cos\phi\, dz\right) = 0. \tag{2.53}$$

Thus a streamfunction can be derived such that

$$\frac{\partial \psi}{\partial \lambda} = \frac{\cos\phi}{R}\int_{-H}^{0} \rho_0 v\, dz = \frac{H\rho_0 \cos\phi}{R}\bar{v} \tag{2.54}$$

$$\frac{\partial \phi}{\partial \phi} = -\frac{1}{R}\int_{-H}^{0} \rho_0 u\, dz \quad = -\frac{H\rho_0}{R}\bar{u}, \tag{2.55}$$

where

$$\bar{u} = \frac{1}{H}\int_{-H}^{0} u\, dz \tag{2.56}$$

$$\bar{v} = \frac{1}{H}\int_{-H}^{0} v\, dz. \tag{2.57}$$

Integrating the momentum equations, (2.38) and (2.39), with respect to z gives

$$\begin{aligned} H\frac{\partial \bar{u}}{\partial t} = {} & Hf\bar{v} - \frac{1}{\rho_0 R\cos\phi}\int_{-H}^{0}\frac{\partial p}{\partial \lambda}dz \\ & + \int_{-H}^{0}\left[-ADV(u) + \frac{uv\tan\phi}{R} + F_\lambda\right]dz \end{aligned} \tag{2.58}$$

$$\begin{aligned} H\frac{\partial \bar{v}}{\partial t} = {} & Hf\bar{u} - \frac{1}{\rho_0 R}\int_{-H}^{0}\frac{\partial p}{\partial \phi}dz \\ & + \int_{-H}^{0}\left[-ADV(v) - \frac{u^2\tan\phi}{R} + F_\phi\right]dz. \end{aligned} \tag{2.59}$$

Using the newly defined streamfunction and the hydrostatic relation

$$p(z) = p_s + g\int_{z}^{0} \rho\, dz', \tag{2.60}$$

where p_s is the surface pressure, then

$$\begin{aligned} -\frac{1}{R\rho_0}\frac{\partial^2 \psi}{\partial \phi \partial t} = {} & \frac{f}{R\rho_0\cos\phi}\frac{\partial \psi}{\partial \lambda} - \frac{H}{R\rho_0\cos\phi}\frac{\partial p_s}{\partial \lambda} \\ & + \int_{-H}^{0}\left[-ADV(u) + \frac{uv\tan\phi}{R} + F_\lambda - \frac{g}{R\rho_0\cos\phi}\int_{z}^{0}\frac{\partial \rho}{\partial \lambda}dz'\right]dz \end{aligned} \tag{2.61}$$

$$\begin{aligned} \frac{1}{R\rho_0\cos\phi}\frac{\partial^2 \psi}{\partial \lambda \partial t} = {} & \frac{f}{R\rho_0}\frac{\partial \psi}{\partial \phi} - \frac{H}{R\rho_0}\frac{\partial p_s}{\partial \phi} \\ & + \int_{-H}^{0}\left[-ADV(v) - \frac{u^2\tan\phi}{R} + F_\phi - \frac{g}{R\rho_0}\int_{z}^{0}\frac{\partial \rho}{\partial \phi}dz'\right]dz. \end{aligned} \tag{2.62}$$

Multiplying these equations by $\frac{R\rho_0 \cos\phi}{H}$ and $\frac{R\rho_0}{H}$ respectively, gives

$$-\frac{\cos\phi}{H}\frac{\partial^2\psi}{\partial\phi\partial t} = \frac{f}{H}\frac{\partial\psi}{\partial\lambda} - \frac{\partial p_s}{\partial\lambda} + \cos\phi FU \tag{2.63}$$

$$\frac{1}{H\cos\phi}\frac{\partial^2\psi}{\partial\lambda\partial t} = \frac{f}{H}\frac{\partial\psi}{\partial\phi} - \frac{\partial p_s}{\partial\phi} + FV, \tag{2.64}$$

where

$$FU = -\frac{R\rho_0}{H}\int_{-H}^{0}\left[ADV(u) - \frac{uv\tan\phi}{R} - F_\lambda + \frac{g}{R\rho_0\cos\phi}\int_z^0 \frac{\partial\rho}{\partial\lambda}dz'\right]dz \tag{2.65}$$

$$FV = -\frac{R\rho_0}{H}\int_{-H}^{0}\left[ADV(v) + \frac{u^2\tan\phi}{R} - F_\phi + \frac{g}{R\rho_0}\int_z^0 \frac{\partial\rho}{\partial\phi}dz'\right]dz. \tag{2.66}$$

The surface pressure is then eliminated by cross-differentiating (2.63) and (2.64) and subtracting the former from the latter, resulting in

$$\frac{\partial}{\partial t}\left[\frac{\partial}{\partial\lambda}\left(\frac{1}{H\cos\phi}\frac{\partial\psi}{\partial\lambda}\right) + \frac{\partial}{\partial\phi}\left(\frac{\cos\phi}{H}\frac{\partial\psi}{\partial\phi}\right)\right]$$
$$= \frac{\partial\psi}{\partial\phi}\frac{\partial}{\partial\lambda}\left(\frac{f}{H}\right) - \frac{\partial\psi}{\partial\lambda}\frac{\partial}{\partial\phi}\left(\frac{f}{H}\right) + \frac{\partial}{\partial\lambda}(FV) - \frac{\partial}{\partial\phi}(\cos\phi FU). \tag{2.67}$$

This equation is Poisson-like, involving difficulties with boundary conditions and complicated topography. See [36] for a description of an instability that can occur with rough bottom topography. The boundary condition for the streamfunction requires integrals around the ocean land masses as described in [11]. This boundary condition procedure is non-local and negatively affects the scalability and efficiency of the model on distributed memory computer architectures.

The total velocity may be expressed as

$$(u, v) = (\bar{u}, \bar{v}) + (\hat{u}, \hat{v}), \tag{2.68}$$

with the $\bar{u}$ and $\bar{v}$ components being predicted by (2.54), (2.55), and (2.67). To predict $\hat{u}$ and $\hat{v}$, the momentum equations, (2.38) and (2.39), are used with the hydrostatic relation

(2.60), where the surface pressure, p_s, is temporarily set to zero to give

$$\frac{\partial u'}{\partial t} + ADV(u) - \frac{uv\tan\phi}{R} - fv = -\frac{g}{R\rho_0\cos\phi}\frac{\partial}{\partial\lambda}\left(\int_z^0 \rho\, dz\right) + F_\lambda \tag{2.69}$$

$$\frac{\partial v'}{\partial t} + ADV(v) - \frac{u^2\tan\phi}{R} - fu = -\frac{g}{R\rho_0}\frac{\partial}{\partial\phi}\left(\int_z^0 \rho\, dz\right) + F_\phi. \tag{2.70}$$

Now u' and v' differ from u and v because of the absence of the part of the pressure gradient due to surface pressure. However, the error due to not including the surface pressure only influences the depth mean of u' and v'. But $\hat{u}$ and $\hat{v}$ have no depth mean, so they can be obtained from u' and v' by subtracting out the depth mean, eliminating the error from the lack of surface pressure. So to determine $\hat{u}$ and $\hat{v}$ the depth means are subtracted from the velocities calculated, i.e.

$$(\hat{u}, \hat{v}) = (u' - \bar{u}', v' - \bar{v}'), \tag{2.71}$$

where $\bar{u}'$ and $\bar{v}'$ are the depth mean values of u' and v'. Thus the velocities have been split into two parts: one part that is independent of depth, known as the barotropic velocity, and another part that is the deviation from the depth mean, known as the baroclinic velocity.

2.2.3 Solving the Free-Surface System – Explicit Method

As mentioned above, allowing surface gravity waves leads to the need for a much shorter time-step. However, as described below, the barotropic component of the flow can be split from the baroclinic. The former is advanced with a smaller timestep, while the more slowly varying terms of the latter can be advanced with the larger timestep. This subcycling strategy can be computationally competitive with the barotropic streamfunction, which generally requires many iterations of a relaxation or conjugate gradient solver.

And as mentioned above, on distributed memory parallel architectures the streamfunction solve involves non-local communication, which can lead to poor performance on large numbers of processors, while the explicit free-surface method only requires local communication. Thus the free-surface approach is preferred for running on massively parallel computers.

The following is a method detailed by Killworth [37]. The hydrostatic relation is defined for the free-surface case as

$$p = p_s + \int_z^0 g\rho \, dz', \tag{2.72}$$

where p_s is the pressure at $z = 0$,

$$p_s = \rho_0 g \eta(\lambda, \phi, t). \tag{2.73}$$

The velocities are still split into barotropic and baroclinic modes,

$$(u, v) = (\bar{u}, \bar{v}) + (\hat{u}, \hat{v}), \tag{2.74}$$

but now the barotropic velocities, $(\bar{u}, \bar{v})$, are defined as depth integrals of the velocities,

$$\bar{u} = \int_{-H}^{\eta} u \, dz \tag{2.75}$$

$$\bar{v} = \int_{-H}^{\eta} v \, dz. \tag{2.76}$$

As with the rigid lid solution, the continuity equation, (2.37), is integrated from the bottom to the surface to give

$$\eta_t + \frac{1}{R \cos\phi} \left[\frac{\partial \bar{u}}{\partial \lambda} + \frac{\partial}{\partial \phi} (\cos\phi \bar{v}) \right] = 0. \tag{2.77}$$

Likewise, integrating the momentum equations and using the boundary conditions on w

gives

$$\bar{u}_t - f\bar{v} = -\frac{gH}{R\cos\phi}\frac{\partial\eta}{\partial\lambda} + X \tag{2.78}$$

$$\bar{v}_t + f\bar{u} = -\frac{gH}{R}\frac{\partial\eta}{\partial\phi} + Y, \tag{2.79}$$

where X and Y are given by

$$\begin{aligned} X = &-\frac{1}{R\cos\phi}\frac{\partial}{\partial\lambda}\int_{-H}^{\eta} u^2\,dz - \frac{1}{R}\frac{\partial}{\partial\phi}\int_{-H}^{\eta} uv\,dz \\ &- \frac{1}{R\cos\phi}\int_{-H}^{\eta} dz\int_{z}^{0} g\frac{\partial\rho}{\partial\lambda}\,dz' + \int_{-H}^{\eta} F_\lambda \end{aligned} \tag{2.80}$$

and

$$\begin{aligned} Y = &-\frac{1}{R\cos\phi}\frac{\partial}{\partial\lambda}\int_{-H}^{\eta} uv\,dz - \frac{1}{R}\frac{\partial}{\partial\phi}\int_{-H}^{\eta} v^2\,dz \\ &- \frac{1}{R}\int_{-H}^{\eta} dz\int_{z}^{0} g\frac{\partial\rho}{\partial\phi}\,dz' + \int_{-H}^{\eta} F_\phi. \end{aligned} \tag{2.81}$$

The terms in X and Y comprise the density forcing, friction, and the nonlinear interactions. All of the terms are assumed to vary slowly or be weak for ocean flows, so that they can be used as forcing terms for the more rapidly evolving barotropic flow.

Thus, the baroclinic velocities are integrated forward in time with (2.69), (2.70), and (2.71). Then, assuming that X and Y are constant, the barotropic equations, (2.77), (2.78), and (2.79), are stepped forward with a smaller time step that satisfies stability for the surface waves until they have reached the same time as the baroclinic equations. Further details and consequences can be found in [37].

2.3 Discretizing the Equations

Using the methods detailed in [11], the equations of the explicit free-surface method described above will now be discretized for solution in the model. First the arrangement of variables on the grid and the specification of topography are defined. Next the general

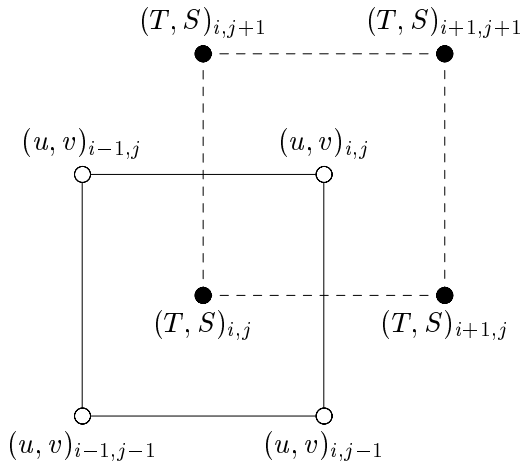

Figure 2.1: The arrangement of variables in the horizontal

discretization operators and methods are described. Then the specific discretizations of each part of the equations are detailed, with separate sections for the tracer transport, baroclinic momentum, and baroclinic momentum equations. Another section describes a numerical problem and its associated solution with a filter. Temporal discretization methods are outlined next, as various methods are used in the model.

2.3.1 Spatial Arrangement of Variables

The arrangement of the variables on the grid in the horizontal is shown in Figure 2.1. This particular arrangement is known as a B-grid (see [2]). It is the most commonly used grid structure in ocean models, but other alternatives have been investigated. Some of the work mentioned in Section 1.2 has been done with unstaggered grids. Reasons for choosing this placement of the model variables include reduction of the truncation error and the relative accuracy of wave speed with the model equations compared with other choices.

Because of this spatial separation of variables, quantities are said to be on the "tracer"

or "velocity" grid. If a quantity $q_{i,j,k}$ is defined on the "tracer" grid, it is located at the T-grid cell with indices (i, j, k). Likewise if $q_{i,j,k}$ is defined in the "velocity" grid, it is located at the U-grid cell with indices (i, j, k), which is located northeast of the T-grid cell with the same indices.

Other quantities of interest will be defined on the faces of the cells. The indexing convention adopted here will be to label the north, east, and bottom faces of a cell with the indices of the cell center. Thus a quantity, q, defined on the north face of the tracer cell $T_{i,j,k}$ is written as $q_{i,j,k}$, and the same quantity on the southern face is written as $q_{i,j-1,k}$. This avoids the use of half indices when writing such expressions.

The vertical arrangement of variables is shown in Figure 2.2. The vertical coordinate is depth, in contrast to some ocean models which use pressure coordinates in the vertical. Level thicknesses are not constant but usually increase with depth. The purpose of the nonconstant vertical resolution is to place more grid levels in the surface regions where gradients are higher and there is more variability. See [69] for details on the implications of the nonconstant, or "stretched", vertical resolution.

All model quantities except the vertical velocity are located at the points marked T in Figure 2.2. Note that these points are not precisely cell centered but lie above the cell center. Rather, the vertical velocity points, marked w in the figure, lie centered between the T points. The opposite method has been used in the past, but the method described places quantities to be advected vertically, which are given by an average of the two vertical levels, coincident with the advecting velocity. See [50] for more discussion of the two options.

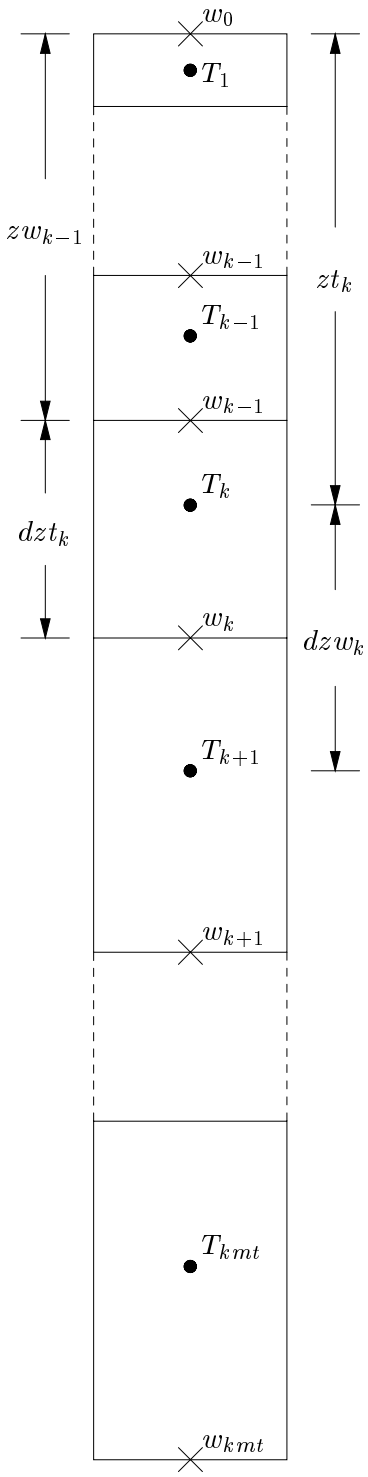

Figure 2.2: The arrangement of variables in the vertical

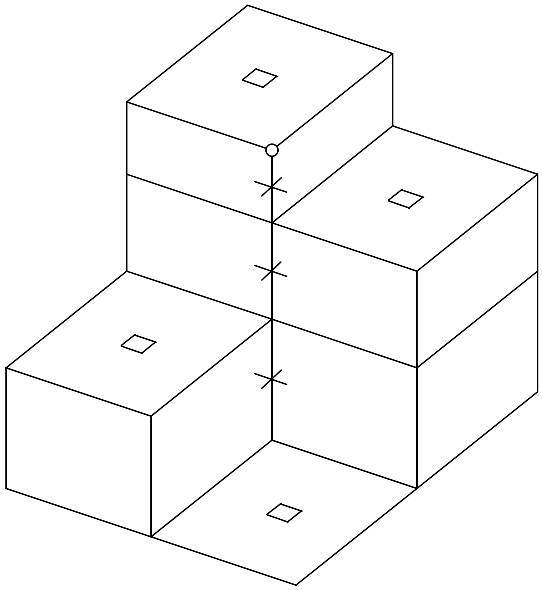

Figure 2.3: An example of bottom topography. The cells shown are tracer land cells. The squares show the locations of vertical velocities at the bottoms of the tracer grid columns. The open circle indicates the location of the vertical velocity at the bottom of the velocity grid column. Cross marks indicate velocity grid land points.

2.3.2 Specification of Topography

The topography of the model domain is given by the number of vertical levels at each horizontal location on the tracer grid. Since the vertical coordinate is depth, this is determined by the depth of the ocean bottom at each horizontal tracer grid location. This information is specified in a file that is read into model arrays on initialization. Subsequently, the number of vertical levels at each location on the velocity grid is determined by the minimum number of levels at the surrounding tracer grid locations.

Figure 2.3 illustrates the relationship between tracer and velocity land cells. The velocities at the cross marks must be zero, as they are located on material surfaces. This brings up an important point. When a tracer grid cell is said to be land, the whole cell is land, and the cell boundaries are material surfaces. However, when a velocity grid cell is said to be land, this means that there is a material surface at the location of the cell

center. The whole cell will be land only in the case where all four neighboring tracer grid cells are also land.

2.3.3 Discrete Spatial Operators

A number of common operations will be used in differencing the equations. The operations of averaging and differencing are defined here. Averaging is given by

$$\overline{q_{i,j,k}}^{\lambda} = \frac{q_{i+1,j,k} + q_{i,j,k}}{2} \tag{2.82}$$

$$\overline{q_{i,j,k}}^{\phi} = \frac{q_{i,j+1,k} + q_{i,j,k}}{2} \tag{2.83}$$

$$\overline{q_{i,j,k}}^{z} = \frac{q_{i,j,k+1} + q_{i,j,k}}{2}. \tag{2.84}$$

There are two forms of differences used, the first defined by

$$\delta_{\lambda}\left(q_{i,j,k}\right) = \frac{q_{i+1,j,k} - q_{i,j,k}}{R\,\Delta\lambda_i} \tag{2.85}$$

$$\delta_{\phi}\left(q_{i,j,k}\right) = \frac{q_{i,j+1,k} - q_{i,j,k}}{R\,\Delta\phi_j} \tag{2.86}$$

$$\delta_{z}\left(q_{i,j,k}\right) = \frac{q_{i,j,k} - q_{i,j,k+1}}{\Delta z_k}, \tag{2.87}$$

and the alternate differencing operations defined by

$$\Delta_{\lambda}\left(q_{i,j,k}\right) = q_{i+1,j,k} - q_{i,j,k} \tag{2.88}$$

$$\Delta_{\phi}\left(q_{i,j,k}\right) = q_{i,j+1,k} - q_{i,j,k} \tag{2.89}$$

$$\Delta_{z}\left(q_{i,j,k}\right) = q_{i,j,k} - q_{i,j,k+1}. \tag{2.90}$$

When q is a tracer grid quantity, then $R\Delta\lambda_i$ is given by dxu_i and $R\Delta\phi_j$ is given by dyu_j. Likewise, when q is a velocity grid quantity, then $R\Delta\lambda_i$ is given by dxt_i and $R\Delta\phi_j$ is given by dyt_j.

Note that, because of the stagger of the variables on the grid, it is important to

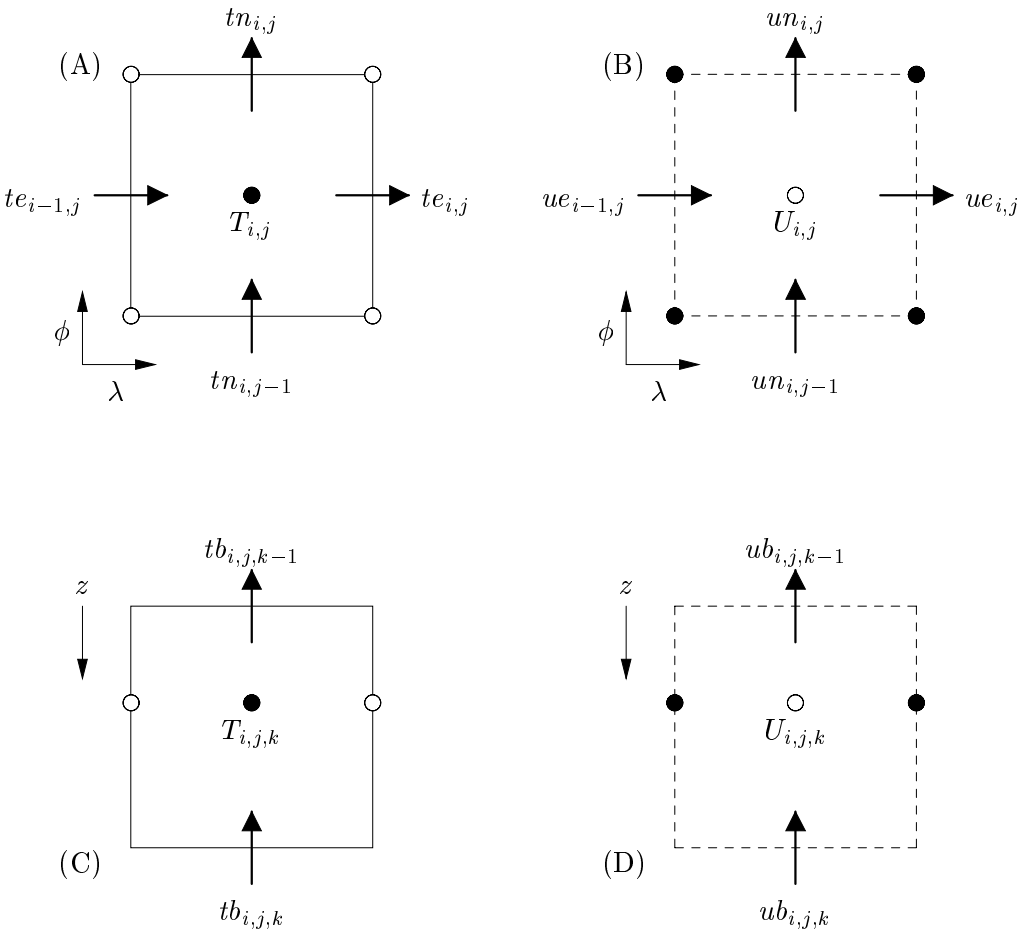

Figure 2.4: Location of some "flux" quantities

know where a quantity is defined and where the result of an operation on that quantity is defined. A few examples are given by the following:

- The zonal average of a tracer grid quantity, $q_{i,j,k}$, given by the expression $\overline{q_{i,j,k}}^{\lambda}$, is defined on the east face of the tracer cell.
- The meridional difference of a velocity grid quantity, $q_{i,j,k}$, given by the expression $\delta_{\phi}\left(q_{i,j,k}\right)$, is defined on the north face of the velocity cell.
- Both the operations $\delta_{\lambda}\left(\overline{q_{i,j,k}}^{\phi}\right)$ and $\delta_{\phi}\left(\overline{q_{i,j,k}}^{\lambda}\right)$, where q is a tracer grid quantity, give quantities defined at the center of the velocity grid cell.
- Both the operations $\delta_{\lambda}\left(\overline{q_{i,j,k}}^{\phi}\right)$ and $\delta_{\phi}\left(\overline{q_{i,j,k}}^{\lambda}\right)$, where q is a velocity grid quantity, give quantities defined at the center of the tracer grid cell.

2.3.4 Flux Form

Because of the Boussinesq approximation of Section 2.1.3, we have $\nabla \cdot \mathbf{u} = 0$, which allows

$$\nabla \cdot (a\mathbf{u}) = a\nabla \cdot \mathbf{u} + \mathbf{u} \cdot \nabla a \tag{2.91}$$

to be reduced to

$$\nabla \cdot (a\mathbf{u}) = \mathbf{u} \cdot \nabla a. \tag{2.92}$$

So the advection operator given by (2.45) can be rewritten as

$$ADV(a) = \frac{1}{R\cos\phi}\frac{\partial(ua)}{\partial\lambda} + \frac{1}{R}\frac{\partial(va)}{\partial\phi} + \frac{\partial(wa)}{\partial z}. \tag{2.93}$$

This "flux form" of the operator highlights its conservation properties by casting it into the divergence of a quantity and will be especially important later in calculating quantities

at the interfaces between grids. In the following sections detailing the discrete form of the equations, most terms will be cast into operations on "flux quantities", so that the transition to the description of the methods used at grid interfaces will be more natural.

2.3.5 Tracer Transport Equations

Using the definitions and forms given above, the discretization of the tracer equations, (2.42) and (2.43) are given here. The methodology will be to break the equations down into general terms which will be defined in more and more specific terms following. The tracer equations can be written as

$$\delta_\tau(t_{i,j,k,\tau}) = ADV(t_{i,j,k,\tau}) + DIFF(t_{i,j,k,\tau-1}), \tag{2.94}$$

where t is any tracer, gives the discrete time rate of change as a sum of an advective and a diffusive term. Here the diffusive term is lagged in time, i.e. the diffusive operator uses the values at the previous timestep. This is generally the case, though not always. The time discretization of this equation and the general methods of time discretization used in the model are discussed in Section 2.3.9.

The advective term will be written in terms of the difference of fluxes into and out of the cell, as described above. First the advective velocities are defined. They are located at the east, north, and bottom faces of tracer grid cells.

$$\text{AdvVel\_Te}_{i,j,k,\tau} = \frac{1}{dyt_j} \cdot \overline{dyu_{j-1} \cdot u_{i,j-1,k,\tau}}^{\phi} \tag{2.95}$$

$$\text{AdvVel\_Tn}_{i,j,k,\tau} = \cos\phi_j^U \cdot \overline{v_{i-1,j,k,\tau}}^{\lambda} \tag{2.96}$$

$$\begin{aligned}\text{AdvVel\_Tb}_{i,j,k,\tau} = &-\frac{1}{\cos\phi_j^T} \times \\ &\sum_{k'=k+1}^{k_{\max}} \left[\delta_\lambda\left(\text{AdvVel\_Te}_{i-1,j,k',\tau}\right) + \delta_\phi\left(\text{AdvVel\_Tn}_{i,j-1,k',\tau}\right)\right] \cdot dzt_k\end{aligned} \tag{2.97}$$

Using these velocities, the advective fluxes of tracer quantities are defined. They are also located on the east, north, and bottom faces of tracer grid cells.

$$\text{AdvFlux\_Te}_{i,j,k,\tau} = \text{AdvVel\_Te}_{i,j,k,\tau} \cdot \overline{t_{i,j,k,\tau}}^{\lambda} \tag{2.98}$$

$$\text{AdvFlux\_Tn}_{i,j,k,\tau} = \text{AdvVel\_Tn}_{i,j,k,\tau} \cdot \overline{t_{i,j,k,\tau}}^{\phi} \tag{2.99}$$

$$\text{AdvFlux\_Tb}_{i,j,k,\tau} = \text{AdvVel\_Tb}_{i,j,k,\tau} \cdot \overline{t_{i,j,k,\tau}}^{z} \tag{2.100}$$

Then for each tracer cell, the time tendency of the tracer quantity due to advection is calculated with differences of these fluxes, with the resulting quantity defined at the centers of tracer grid cells.

$$\begin{aligned} ADV(t_{i,j,k,\tau}) = \frac{1}{\cos\phi_j^T} &\left[\delta_\lambda\left(\text{AdvFlux\_Te}_{i-1,j,k,\tau}\right) + \delta_\phi\left(\text{AdvFlux\_Tn}_{i,j-1,k,\tau}\right)\right] \\ &+ \delta_z\left(\text{AdvFlux\_Tb}_{i,j,k-1,\tau}\right) \end{aligned} \tag{2.101}$$

For tracer diffusion, again the terms are written as the difference of fluxes into and out of the cell. The diffusive fluxes of tracer quantities are defined on the east, north, and bottom faces of the tracer grid cells.

$$\text{DiffFlux\_Te}_{i,j,k,\tau} = A_{Hj} \cdot \delta_\lambda\left(t_{i,j,k,\tau}\right) \tag{2.102}$$

$$\text{DiffFlux\_Tn}_{i,j,k,\tau} = A_{Hj}\cos\phi_j^U \cdot \delta_\phi\left(t_{i,j,k,\tau}\right) \tag{2.103}$$

$$\text{DiffFlux\_Tb}_{i,j,k,\tau} = \kappa_{Hk} \cdot \delta_z\left(t_{i,j,k,\tau}\right) \tag{2.104}$$

Then, for each tracer cell, the time tendency of the tracer quantity due to diffusion is calculated with differences of these fluxes, with the resulting quantity defined at the centers of the tracer grid cells.

$$\begin{aligned} DIFF(t_{i,j,k,\tau}) = \frac{1}{\cos^2\phi_j^T}&\delta_\lambda\left(\text{DiffFlux\_Te}_{i-1,j,k,\tau}\right) \\ &+ \frac{1}{\cos\phi_j^T}\delta_\phi\left(\text{DiffFlux\_Tn}_{i,j-1,k,\tau}\right) \\ &+ \delta_z\left(\text{DiffFlux\_Tb}_{i,j,k-1,\tau}\right) \end{aligned} \tag{2.105}$$

The use of the hydrostatic approximation, (2.34), requires the use of a vertical convection scheme to ensure static stability. This is accomplished by an explicit convection scheme which operates whenever there is a gravitational instability given by

$$\delta_z \left(\rho_{i,j,k}\right) < 0. \tag{2.106}$$

The convection scheme operates by an iterative process which mixes pairs of adjacent levels which are unstable. The process is outlined is the following algorithm:

Algorithm 2.3.1. Convective Adjustment
for $m = 1$ to n
for $s = 1$ to 2
if s=1 **then**
Compute densities, $rho_{i,j,k}$, referenced to the same pressure level for each of the level pairs 1&2, 3&4, etc.
else
Compute densities, $rho_{i,j,k}$, referenced to the same pressure level for each of the level pairs 2&3, 4&5, etc.
end if
for $k = s$ to $kmax - 1$
if $\rho_{i,j,k} > \rho_{i,j,k+1}$ **then**
$T_{i,j,k} \Leftarrow \frac{T_{i,j,k} dzt_k + T_{i,j,k+1} dzt_{k+1}}{dzt_k + dzt_{k+1}}$
$T_{i,j,k+1} \Leftarrow T_{i,j,k}$
end if
end for
end for
end for

In this algorithm n is a parameter that is adjustable. Higher values mix the column more thoroughly, while lower values reduce the computational expense. Since this scheme is not guaranteed to remove all instabilities, using a value that is too low for n may leave some gravitational instability in the solution.

2.3.6 Baroclinic Momentum Equations

The baroclinic momentum equations, (2.69) and (2.70), can similarly be written in the form

$$\begin{aligned}\delta_\tau(u_{i,j,k,\tau}) = ADV(u_{i,j,k,\tau}) + COR(v_{i,j,k,\tau'}) + MET(u_{i,j,k,\tau}) \\ + GRAD^\lambda(p_{i,j,k,\tau''}) + VISC(u_{i,j,k,\tau-1})\end{aligned} \tag{2.107}$$

$$\begin{aligned}\delta_\tau(v_{i,j,k,\tau}) = ADV(v_{i,j,k,\tau}) + COR(u_{i,j,k,\tau'}) + MET(v_{i,j,k,\tau}) \\ + GRAD^\phi(p_{i,j,k,\tau''}) + VISC(v_{i,j,k,\tau-1}),\end{aligned} \tag{2.108}$$

where u and v are the baroclinic velocities. The time discretization of these equations is discussed in Section 2.3.9, along with the meanings of the time indices, τ' and τ''. There are many more terms involved in these equations, and the stagger of the grid places similar quantities in different locations on the grid. However, the general forms are the same.

Starting again with the advective terms, velocities are defined for the advection of momentum. They are defined on the east, north, and bottom faces of velocity grid cells.

$$\mathrm{AdvVel\_Ue}_{i,j,k,\tau} = \overline{u_{i,j,k,\tau}}^{\lambda} \tag{2.109}$$

$$\mathrm{AdvVel\_Un}_{i,j,k,\tau} = \cos\phi^T_{j+1} \cdot \overline{v_{i,j,k,\tau}}^{\phi} \tag{2.110}$$

$$\begin{aligned}\mathrm{AdvVel\_Ub}_{i,j,k,\tau} = \overline{\mathrm{AdvVel\_Tb}_{i,j,k_{\mathrm{bottom}},\tau}}^{\lambda\phi} - \\ \frac{1}{\cos\phi^U_j} \sum_{k'=k+1}^{k_{\max}} \left[\delta_\lambda\left(\mathrm{AdvVel\_Ue}_{i-1,j,k',\tau}\right) + \delta_\phi\left(\mathrm{AdvVel\_Un}_{i,j-1,k',\tau}\right)\right] \cdot dzu_k.\end{aligned} \tag{2.111}$$

Using these velocities, advective fluxes can be written, which are located on the east,

north, and bottom faces of velocity grid cells:

$$\text{AdvFlux\_Ue}_{i,j,k,\tau} = \text{AdvVel\_Ue}_{i,j,k,\tau} \cdot \overline{u_{i,j,k,\tau}}^{\lambda} \tag{2.112}$$

$$\text{AdvFlux\_Un}_{i,j,k,\tau} = \text{AdvVel\_Un}_{i,j,k,\tau} \cdot \overline{u_{i,j,k,\tau}}^{\phi} \tag{2.113}$$

$$\text{AdvFlux\_Ub}_{i,j,k,\tau} = \text{AdvVel\_Ub}_{i,j,k,\tau} \cdot \overline{u_{i,j,k,\tau}}^{z} \tag{2.114}$$

and

$$\text{AdvFlux\_Ve}_{i,j,k,\tau} = \text{AdvVel\_Ue}_{i,j,k,\tau} \cdot \overline{v_{i,j,k,\tau}}^{\lambda} \tag{2.115}$$

$$\text{AdvFlux\_Vn}_{i,j,k,\tau} = \text{AdvVel\_Un}_{i,j,k,\tau} \cdot \overline{v_{i,j,k,\tau}}^{\phi} \tag{2.116}$$

$$\text{AdvFlux\_Vb}_{i,j,k,\tau} = \text{AdvVel\_Ub}_{i,j,k,\tau} \cdot \overline{v_{i,j,k,\tau}}^{z}. \tag{2.117}$$

Then for each velocity cell, the time tendency of the velocities due to advection is calculated with differences of these fluxes, with the resulting quantity defined at the centers of the velocity grid cells.

$$\begin{aligned} ADV(u_{i,j,k,\tau}) = \frac{1}{\cos\phi_j^U} &\left[\delta_\lambda\left(\text{AdvFlux\_Ue}_{i-1,j,k,\tau}\right) + \delta_\phi\left(\text{AdvFlux\_Un}_{i,j-1,k,\tau}\right)\right] \\ &+ \delta_z\left(\text{AdvFlux\_Ub}_{i,j,k-1,\tau}\right) \end{aligned} \tag{2.118}$$

$$\begin{aligned} ADV(v_{i,j,k,\tau}) = \frac{1}{\cos\phi_j^U} &\left[\delta_\lambda\left(\text{AdvFlux\_Ve}_{i-1,j,k,\tau}\right) + \delta_\phi\left(\text{AdvFlux\_Vn}_{i,j-1,k,\tau}\right)\right] \\ &+ \delta_z\left(\text{AdvFlux\_Vb}_{i,j,k-1,\tau}\right) \end{aligned} \tag{2.119}$$

The diffusive fluxes of momentum are also defined on the east, north, and bottom faces of velocity grid cells.

$$\text{DiffFlux\_Ue}_{i,j,k,\tau} = A_{Mj} \cdot \delta_\lambda\left(u_{i,j,k,\tau}\right) \tag{2.120}$$

$$\text{DiffFlux\_Un}_{i,j,k,\tau} = A_{Mj}\cos\phi_{j+1}^T \cdot \delta_\phi\left(u_{i,j,k,\tau}\right) \tag{2.121}$$

$$\text{DiffFlux\_Ub}_{i,j,k,\tau} = \kappa_{Mk} \cdot \delta_z\left(u_{i,j,k,\tau}\right) \tag{2.122}$$

and

$$\text{DiffFlux\_Ve}_{i,j,k,\tau} = A_{Mj} \cdot \delta_\lambda \left(v_{i,j,k,\tau}\right) \tag{2.123}$$

$$\text{DiffFlux\_Vn}_{i,j,k,\tau} = A_{Mj} \cos \phi^T_{j+1} \cdot \delta_\phi \left(v_{i,j,k,\tau}\right) \tag{2.124}$$

$$\text{DiffFlux\_Vb}_{i,j,k,\tau} = \kappa_{Mk} \cdot \delta_z \left(v_{i,j,k,\tau}\right) \tag{2.125}$$

These differ from the expressions given in [50] and others, in that variable grid spacing in the longitudinal direction is not allowed. The viscous terms are much like the tracer diffusion terms, except they contain extra metric terms (see equation (2.46)), and they are defined at the centers of velocity grid cells.

$$\begin{aligned} VISC(u_{i,j,k,\tau}) = & \frac{1}{\cos^2 \phi^U_j} \delta_\lambda \left(\text{DiffFlux\_Ue}_{i-1,j,k,\tau}\right) \\ & + \frac{1}{\cos \phi^U_j} \delta_\phi \left(\text{DiffFlux\_Un}_{i,j-1,k,\tau}\right) \\ & + \delta_z \left(\text{DiffFlux\_Ub}_{i,j,k-1,\tau}\right) + VMET(u_{i,j,k,\tau}) \end{aligned} \tag{2.126}$$

$$\begin{aligned} VISC(v_{i,j,k,\tau}) = & \frac{1}{\cos^2 \phi^U_j} \delta_\lambda \left(\text{DiffFlux\_Ve}_{i-1,j,k,\tau}\right) \\ & + \frac{1}{\cos \phi^U_j} \delta_\phi \left(\text{DiffFlux\_Vn}_{i,j-1,k,\tau}\right) \\ & + \delta_z \left(\text{DiffFlux\_Vb}_{i,j,k-1,\tau}\right) + VMET(v_{i,j,k,\tau}) \end{aligned} \tag{2.127}$$

where

$$\begin{aligned} VMET(u_{i,j,k,\tau}) = & \frac{A_{Mj}(1 - \tan^2 \phi^U_j)}{R^2} u_{i,j,k,\tau} \\ & - \frac{A_{Mj} \sin \phi^U_j}{R\, dxu_j \cos^2 \phi^U_j} \left(v_{i+1,j,k,\tau} - v_{i-1,j,k,\tau}\right) \end{aligned} \tag{2.128}$$

$$\begin{aligned} VMET(v_{i,j,k,\tau}) = & \frac{A_{Mj}(1 - \tan^2 \phi^U_j)}{R^2} v_{i,j,k,\tau} \\ & - \frac{A_{Mj} \sin \phi^U_j}{R\, dxu_j \cos^2 \phi^U_j} \left(u_{i+1,j,k,\tau} - u_{i-1,j,k,\tau}\right) \end{aligned} \tag{2.129}$$

The coriolis terms, which are not present in the tracer transport equations, are of a particularly simple form, as they are merely linear functions of the velocities, and they are located at the centers of velocity cells.

$$COR(u_{i,j,k,\tau}) = 2\Omega \sin \phi_j^U u_{i,j,k,\tau} \tag{2.130}$$

$$COR(v_{i,j,k,\tau}) = 2\Omega \sin \phi_j^U v_{i,j,k,\tau}, \tag{2.131}$$

where τ' will be τ for an explicit treatment of the term and a (possible) mixture of $\tau - 1$ and $\tau + 1$ for implicit treatment, as discussed in Section 2.3.9.

The metric terms, also located at the centers of velocity cells, are given by

$$MET(u_{i,j,k,\tau}) = \frac{u_{i,j,k,\tau} v_{i,j,k,\tau} \tan \phi_j^U}{R} \tag{2.132}$$

$$MET(v_{i,j,k,\tau}) = \frac{u_{i,j,k,\tau}^2 \tan \phi_j^U}{R}. \tag{2.133}$$

The pressure gradient terms are calculated by derivatives of the pressure, which is given by the hydrostatic relation of (2.34). Since pressure is calculated by integrating the density from the surface downward, these terms are sums of the discretized derivatives of the density. They are located at the centers of velocity grid cells. At the first depth level, the pressure gradient is given by

$$GRAD^\lambda(p_{i,j,1,\tau}) = \frac{-g\, dzw_1}{\rho_0 \cos \phi_j^U} \delta_\lambda \left(\overline{\rho_{i,j,1,\tau'}}^{\phi} \right) \tag{2.134}$$

$$GRAD^\phi(p_{i,j,1,\tau}) = \frac{-g\, dzw_1}{\rho_0} \delta_\phi \left(\overline{\rho_{i,j,1,\tau'}}^{\lambda} \right). \tag{2.135}$$

And then at the deeper levels, it is given by

$$GRAD^\lambda(p_{i,j,k,\tau}) = GRAD^\lambda(p_{i,j,1,\tau}) + \frac{-g}{\rho_0 \cos \phi_j^U} \sum_{k'=2}^{k} \delta_\lambda \left(\overline{\rho_{i,j,k'-1,\tau'}}^{\phi,z} \right) dzw_{k'} \tag{2.136}$$

$$GRAD^\phi(p_{i,j,k,\tau}) = GRAD^\phi(p_{i,j,1,\tau}) + \frac{-g}{\rho_0} \sum_{k'=2}^{k} \delta_\phi \left(\overline{\rho_{i,j,k'-1,\tau'}}^{\lambda,z} \right) dzw_{k'}, \tag{2.137}$$

where $\rho_{\tau'}$ may be either ρ_τ or the time averaged quantity $\frac{1}{4}(\rho_{\tau-1} + 2\rho_\tau + \rho_{\tau+1})$.

2.3.7 Barotropic Momentum and Surface Height Equations

The barotropic momentum equations, (2.78) and (2.79), and the surface height equation, (2.77), can be written in discrete form as

$$\delta_\tau(\bar{u}_{i,j,\tau}) = COR(\bar{u}_{i,j,\tau'}) + GRAD^\lambda(\eta_{i,j,\tau}) + X_{i,j,\tau''} \tag{2.138}$$

$$\delta_\tau(\bar{v}_{i,j,\tau}) = -COR(\bar{v}_{i,j,\tau'}) + GRAD^\phi(\eta_{i,j,\tau}) + Y_{i,j,\tau''} \tag{2.139}$$

$$\delta_\tau(\eta_{i,j,\tau}) = GRAD^\lambda(\bar{u}_{i,j,\tau}) + GRAD^\phi(\bar{v}_{i,j,\tau}), \tag{2.140}$$

with each of the forcing terms in the first two equations defined at the centers of velocity grid cells, and the forcing terms of the third equation defined at the centers of tracer grid cells. The time discretization of these equations is discussed in Section 2.3.9, along with the meanings of the time indices, τ' and τ''.

The terms X and Y contain the slowly varying forcing terms, which are assumed constant during the integration of the barotropic equations.

$$X_{i,j,\tau} = \sum_{k=1}^{k_{\max}} \Big[ADV(u_{i,j,k,\tau}) + MET(u_{i,j,k,\tau}) + GRAD^\lambda(p_{i,j,k,\tau}) + VISC(u_{i,j,k,\tau}) \Big] dzu_k \tag{2.141}$$

$$Y_{i,j,\tau} = \sum_{k=1}^{k_{\max}} \Big[ADV(v_{i,j,k,\tau}) + MET(v_{i,j,k,\tau}) + GRAD^\phi(p_{i,j,k,\tau}) + VISC(v_{i,j,k,\tau}) \Big] dzu_k \tag{2.142}$$

The above terms are those defined for the baroclinic equations, (2.118), (2.132), (2.136), and (2.126), respectively, for the X term, and (2.119), (2.133), (2.137), and (2.127), respectively, for the Y term. They are calculated once during the baroclinic advance and are summed and stored for use here in the barotropic advance.

The coriolis terms in the barotropic equations are nearly identical to those in the

baroclinic equations, (2.130) and (2.131).

$$COR(\bar{u}_{i,j,\tau}) = 2\Omega \sin \phi_j^U \cdot \bar{v}_{i,j,\tau} \tag{2.143}$$

$$COR(\bar{v}_{i,j,\tau}) = 2\Omega \sin \phi_j^U \cdot \bar{u}_{i,j,\tau} \tag{2.144}$$

The remaining forcing terms in the barotropic velocity equations are spatial derivatives of the surface height.

$$GRAD^{\lambda}(\eta_{i,j,\tau}) = \frac{gH_{i,j}}{\cos \phi_j^U} \cdot \delta_{\lambda} \left(\text{EtaFlux\_e}_{i-1,j,\tau} \right) \tag{2.145}$$

$$GRAD^{\phi}(\eta_{i,j,\tau}) = gH_{i,j} \cdot \delta_{\phi} \left(\text{EtaFlux\_n}_{i,j-1,\tau} \right), \tag{2.146}$$

where the fluxes are given by

$$\text{EtaFlux\_e}_{i,j,\tau} = \overline{\eta_{i,j,\tau}}^{\phi} \tag{2.147}$$

$$\text{EtaFlux\_n}_{i,j,\tau} = \overline{\eta_{i,j,\tau}}^{\lambda}, \tag{2.148}$$

which are defined at the east and north faces of velocity grid cells.

The surface height equation has forcing terms very much like the second terms in the barotropic velocity equations.

$$GRAD^{\lambda}(\bar{u}_{i,j,\tau}) = \frac{1}{\cos \phi_j^T} \cdot \delta_{\lambda} \left(\text{UFlux}_{i-1,j,\tau} \right) \tag{2.149}$$

$$GRAD^{\phi}(\bar{v}_{i,j,\tau}) = \frac{1}{\cos \phi_j^T} \cdot \delta_{\phi} \left(\text{VFlux}_{i,j-1,\tau} \right), \tag{2.150}$$

where the fluxes are given by

$$\text{UFlux}_{i,j,\tau} = \overline{\bar{u}_{i,j-1,\tau}}^{\phi} \tag{2.151}$$

$$\text{VFlux}_{i,j,\tau} = \cos \phi_j^U \cdot \overline{\bar{v}_{i-1,j,\tau}}^{\lambda}, \tag{2.152}$$

which are defined at the east and north faces of tracer grid cells, respectively.

2.3.8 Surface Height Filter

The discrete equation for the surface height given in Section 2.3.7 often computes a solution with what looks like a "checkerboard" spatial pattern. This part of the solution is not a feature of the physical equations. However, due to the nature of the discrete numerics with the choice of mesh having variables defined at staggered locations, this pattern is persistent in the solution to the discrete equations. That is, once this "checkerboard" component exists in the discrete solution, the primary terms in the discrete equations have no effect on it.

Another way of stating this is to say the "checkerboard" solution in the surface height is in the "null space" of the discrete operators. That is, for a discrete operator, $\mathcal{L}(\eta)$, and a purely "checkerboard" solution, η^C, given by

$$\eta^C_{i,j} = a\,(-1)^{i+j} \tag{2.153}$$

where a is a constant, η^C is in the "null space" of the operator if

$$\mathcal{L}\left(\eta^C\right) = 0. \tag{2.154}$$

Thus, once the solution has this component, the operator has no effect on it.

As a demonstration of this in the barotropic equations, Figure 2.5 shows the surface height field given by (2.153). The surface height appears in the equation (2.138) and (2.139), operated on by the gradient operators $GRAD^{\lambda}(\eta)_{i,j}$ and $GRAD^{\phi}(\eta)_{i,j}$. But these operators, defined by (2.145) and (2.146), calculate spatial differences of fluxes in one horizontal dimension given by spatial averages in the other horizontal dimension. Using

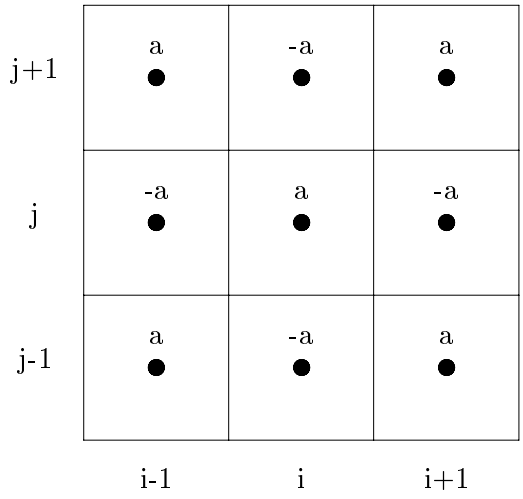

Figure 2.5: An example of a "checkerboard" solution in the surface height.

the example surface height field,

$$\text{EtaFlux\_e}_{i,j} = \frac{a + (-a)}{2} = 0 \tag{2.155}$$

$$\text{EtaFlux\_e}_{i+1,j} = \frac{-a + a}{2} = 0 \tag{2.156}$$

$$\text{EtaFlux\_n}_{i,j} = \frac{a + (-a)}{2} = 0 \tag{2.157}$$

$$\text{EtaFlux\_n}_{i,j+1} = \frac{-a + a}{2} = 0, \tag{2.158}$$

giving zero for the gradient operators, which will be taking differences of a constant. Since the surface height is modified using terms computed from the barotropic velocities, but the barotropic velocities are not influenced by this "checkerboard" solution, the solution is allowed to persist.

Since this solution is a numerical artifact, it should be removed before it grows in size and generates roundoff errors which can affect the solution. As discussed in [18] and [37], a filter can be applied to the surface height to selectively damp this mode. First land

masks are defined for both the tracer cells and the velocity cells.

$$\text{mask}^T_{i,j} = \begin{cases} 0 & \text{if } i,j \text{ is land} \\ \\ 1 & \text{if } i,j \text{ is water} \end{cases} \tag{2.159}$$

$$\text{mask}^U_{i,j} = \min\left(\text{mask}^T_{i,j}, \text{mask}^T_{i+1,j}, \text{mask}^T_{i,j+1}, \text{mask}^T_{i+1,j+1}\right) \tag{2.160}$$

Then fluxes can be defined at the east and north faces of the tracer grid cells,

$$\text{DelPlusFlux\_e}(\eta_{i,j}) = \cos^T_j \, \text{mask}^T_{i,j}\text{mask}^T_{i+1,j} \cdot \Delta_\lambda \left(\eta_{i,j}\right) \tag{2.161}$$

$$\text{DelPlusFlux\_n}(\eta_{i,j}) = \cos^U_j \, \text{mask}^T_{i,j}\text{mask}^T_{i,j+1} \cdot \Delta_\phi \left(\eta_{i,j}\right), \tag{2.162}$$

which allow the definition of an operator similar to a laplacian,

$$\text{DelPlus}(\eta_{i,j}) = \Delta_\lambda \left(\text{DelPlusFlux\_e}(\eta_{i-1,j})\right) + \Delta_\phi \left(\text{DelPlusFlux\_n}(\eta_{i,j-1})\right). \tag{2.163}$$

Then other fluxes are defined at the northeast and northwest corners of the tracer grid cells,

$$\text{DelCrossFlux\_ne}(\eta_{i,j}) = \cos^U_j \, \text{mask}^U_{i,j} \cdot \left(\eta_{i+1,j+1} - \eta_{i,j}\right) \tag{2.164}$$

$$\text{DelCrossFlux\_nw}(\eta_{i,j}) = \cos^U_j \, \text{mask}^U_{i-1,j} \cdot \left(\eta_{i-1,j+1} - \eta_{i,j}\right), \tag{2.165}$$

which allow the definition of another operator which is similar to the "DelPlus" operator of (2.163), but which is oriented such that it is logically orthogonal to it,

$$\begin{aligned} \text{DelCross}(\eta_{i,j}) = \frac{1}{2} \big[&\text{DelCrossFlux\_ne}(\eta)_{i,j} - \text{DelCrossFlux\_ne}(\eta)_{i-1,j-1} + \\ &\text{DelCrossFlux\_nw}(\eta)_{i,j} - \text{DelCrossFlux\_nw}(\eta)_{i+1,j-1} \big]. \end{aligned} \tag{2.166}$$

Using these fluxes, an operator can be defined by

$$\text{CheckerFilt}(\eta_{i,j}) = \eta_{i,j} + \frac{\alpha}{\cos\phi^T_j \, dyt_j} \left[\text{DelPlus}(\eta)_{i,j} - \text{DelCross}(\eta)_{i,j}\right], \tag{2.167}$$

where α is a coeficient determining the strength of the filter. When applied to the surface height, (2.167) selectively filters out the "checkerboard" solution.

To see the effects of this filtering operator, apply it to the previous example solution shown in Figure 2.5. Assuming all values of the masks are one, and using cartesian coordinates, so that $\cos\phi_j$ is constant in j, for simplicity. Then

$$\text{DelPlus}(\eta_{i,j}) = [(-a) - a] - [a - (-a)] + [(-a) - a] - [a - (-a)] = -8a \tag{2.168}$$

$$\text{DelCross}(\eta_{i,j}) = \frac{1}{2}[(a - a) - (a - a) + (a - a) - (a - a)] = 0, \tag{2.169}$$

so that

$$\text{CheckerFilt}(\eta_{i,j}) = a + \frac{\alpha}{dyt_j}(-8a - 0). \tag{2.170}$$

In the absence of variation in the meridional spacing, the use of $\alpha = dyt/8$ will result in those cells with the value a becoming zero, and those with value $-a$ becoming zero as well (by simply reversing all signs in the calculation), thus eliminating the checkerboard solution entirely.

Effectively, this filter is approximating the curvature of the solution twice, along two directions as nearly orthogonal as possible, then modifying the solution proportional to the difference of the two estimates. Thus for solutions with fairly smooth local curvature, that is, curvature which is nearly the same when calculated along the two different directions, the filter has very little effect. For example, when there is locally constant curvature in only one direction, this filter has zero effect. Yet any "checkerboard" component to the solution can be eliminated almost entirely. See [18] for more details of the numerical properties and a discussion of more general filters. Refer to the sections of this work describing the model runs for detail of where and how this filter is used.

2.3.9 Temporal Discretization

In the previous sections, discrete operators have been written which form the right-hand side of equations with the general form,

$$\delta_\tau(q_{i,j,k,\tau}) = \mathcal{L}(q_{i,j,k,\tau}). \tag{2.171}$$

Now the discretization of the left-hand side must be defined. In the current model, the discrete time operator, δ_τ, may be a function of both the current timestep, τ, the previous timestep, $\tau - 1$, and the next timestep, $\tau + 1$, depending on the time-stepping procedure used. The particular procedures used in this work will be outlined here.

Leapfrog

The main method of time advance in the present model is that of centered differences in time, or "leapfrog" time stepping as it is sometimes called. Using a centered difference in time on the left-hand side of (2.171) gives

$$\frac{q_{i,j,k,\tau+1} - q_{i,j,k,\tau-1}}{2\,dt} = \mathcal{L}(q_{i,j,k,\tau}), \tag{2.172}$$

where dt is the time step. Solving for the the level $\tau + 1$,

$$q_{i,j,k,\tau+1} = q_{i,j,k,\tau-1} + 2\,dt\,\mathcal{L}(q_{i,j,k,\tau}). \tag{2.173}$$

Thus the solution at the next time step, $\tau + 1$, is obtained by modifying the solution at the previous time step, $\tau - 1$, based on discrete operations performed on the solution at the current time step, τ. One advantage of this method is accuracy of second-order (see Section A.2) with only one evaluation of the right-hand side of (2.171) per time step. Disadvantanges include the need to provide space for the model variables at three time levels and the potential for the solution to decouple into separate solutions in time

(see [31]). Because of the latter problem, the following method is used at intervals in conjuction with leapfrog time stepping.

Euler Backward

This method is a two-step process consisting of a prediction step combined with a correction step. The prediction step is a forward in time differencing of (2.171) from the current time level, giving a prediction of the value of $q_{i,j,k,\tau+1}$ of

$$\frac{q_{i,j,k,\tau'} - q_{i,j,k,\tau}}{dt} = \mathcal{L}(q_{i,j,k,\tau}), \tag{2.174}$$

giving

$$q_{i,j,k,\tau'} = q_{i,j,k,\tau} + dt\, \mathcal{L}q_{i,j,k,\tau}. \tag{2.175}$$

Then this estimate is used in a backward differencing to achieve a corrected value,

$$\frac{q_{i,j,k,\tau+1} - q_{i,j,k,\tau}}{dt} = \mathcal{L}(q_{i,j,k,\tau'}), \tag{2.176}$$

giving

$$q_{i,j,k,\tau+1} = q_{i,j,k,\tau} + dt\, \mathcal{L}(q_{i,j,k,\tau'}). \tag{2.177}$$

This two step method requires two computations of the right-hand side of (2.171) as well as the same storage requirements as the leapfrog method. It is first-order accurate, but the steady-state solution contains no numerical dissipation (again see Section A.2), unlike the first-order forward Euler differencing. Unlike the leapfrog method, the Euler backward time stepping does not suffer from the temporal splitting of the solution but damps this computational mode.

General Time Stepping Process

The above two time stepping methods are used together in the model to provide efficiency without the problems that the leapfrog method alone could exhibit. The leapfrog method is used, while an Euler backward time step is made periodically to damp the computational mode that can arise in the leapfrog scheme. Past experience of modelers has shown that a frequency of one mixing step per 17 leapfrog steps gives good results. Also, for the first time step of the model, when the initial solution is only known at one time level, an Euler backward time step is used to initiate the integration.

Implicit Coriolis Terms

The coriolis terms in the equations may be treated either explicitly, implicitly, or as a mixture. That is, when calculating the coriolis terms, either the value of the variables at the current or previous timesteps, the value at the next timestep (which is what is sought), or a mixture of the two can be used. To show how this is done in general, take a simplified set of equations which have coriolis terms,

$$U^{\tau+1} = U^{\tau} + k\left\{\mathcal{L}\left(U^{\tau}\right) + f\left[(1-\alpha)V^{\tau} + \alpha V^{\tau+1}\right]\right\} \tag{2.178}$$

$$V^{\tau+1} = V^{\tau} + k\left\{\mathcal{L}\left(V^{\tau}\right) - f\left[(1-\alpha)U^{\tau} + \alpha U^{\tau+1}\right]\right\}, \tag{2.179}$$

where $\mathcal{L}$ is a finite difference operator, k is the timestep, f is the coriolis parameter, α is the fraction of the coriolis term to compute implicitly, and forward in time differencing has been assumed for simplicity. Take (2.179) and substitute it into (2.178) for $V^{\tau+1}$, giving

$$\begin{aligned} U^{\tau+1} = U^{\tau} &+ k\left[\mathcal{L}\left(U^{\tau}\right) + (1-\alpha)fV^{\tau}\right] + \alpha fkV^{\tau} \\ &+ \alpha fk^2\mathcal{L}\left(V^{\tau}\right) - \alpha(1-\alpha)f^2k^2U^{\tau} - \alpha^2f^2k^2U^{\tau+1} \end{aligned} \tag{2.180}$$

or

$$U^{\tau+1} = \frac{\left[1-\alpha(1-\alpha)f^2k^2\right]U^\tau + k\mathcal{L}\left(U^\tau\right) + fk\left[V^\tau + \alpha k\mathcal{L}\left(V^\tau\right)\right]}{1+\alpha^2f^2k^2}. \tag{2.181}$$

Likewise, by substituting (2.178) into (2.179) for $U^{\tau+1}$,

$$V^{\tau+1} = \frac{\left[1-\alpha(1-\alpha)f^2k^2\right]V^\tau + k\mathcal{L}\left(V^\tau\right) - fk\left[U^\tau + \alpha k\mathcal{L}\left(U^\tau\right)\right]}{1+\alpha^2f^2k^2}. \tag{2.182}$$

2.4 Parallel Implementation

The parallelization of the LLNL ocean model is based on two-dimensional domain decomposition. All points in the vertical column at a given latitude and longitude are calculated by the same processor. This choice reflects the very different nature of the vertical dimension when compared to the horizontal dimensions. Processes such as convection and calculations of quantities such as pressure take place over all of the points in the vertical column. No such analogous processes or calculations occur over all of the points of the horizontal dimensions, except for the filtering of variables at high latitudes, which because of this property presents difficulties for effective parallelization. A decomposition in which all grid cells in the vertical are handled by the same processor is therefore desirable to minimize interprocessor communication. Also, since the barotropic components of the model equations are two-dimensional, a two-dimensional decomposition simplifies the barotropic/baroclinic split as well.

When splitting up the domain over a given number of processors, the subdomains are assigned in a regular cartesian manner, without considering which columns contain all land cells and which have water cells. An example processor layout is shown in Figure 2.6. This type of decomposition yields a simple, logical arrangement in which each subdomain has either one or no neighbors on each of its north, south, east, and west borders, and

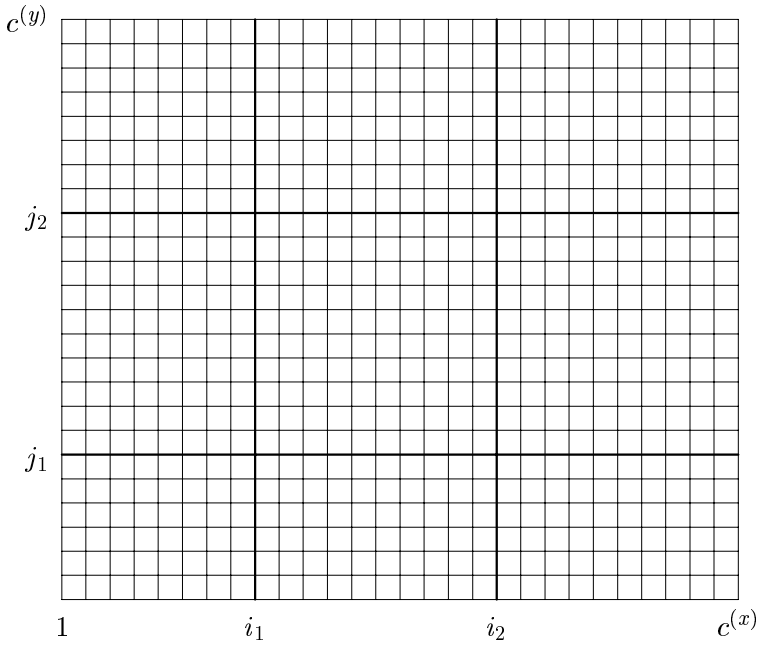

Figure 2.6: An example of regular two-dimensional domain decomposition for nine subdomains.

those neighbors have the same domain widths and heights, in the cases of north/south and east/west neighbors, respectively. Each subdomain contains a one-cell border around all sides, referred to as "ghost" zones, in addition to the cells assigned to it by the decomposition. These ghost cells correspond to the nearest cells of each neighboring subdomain and are used for computation convenience and efficiency.

Unlike the Figure 2.6, the choice of how to divide the domain—that is, how to choose the indices i_n and j_n—is made such that each subdomain has as close to the average number of cells as possible.

Algorithm 2.4.1. Regular one-dimensional domain decomposition
Given:
s {a number of subdomains over which to divide the domain}
c {the number of cells in the domain}
Calculate:
$d_{\text{avg}} \Leftarrow c \div s$ {integer division}
$r \Leftarrow c \bmod s$ {remainder}
if $r \neq 0$ **then** {domain does not divide evenly}
do $d_n \Leftarrow c_{\text{avg}} + 1$ **for** $n = 1$ to r
end if
do $d_n \Leftarrow c_{\text{avg}}$ **for** $n = r + 1$ to s
Then each subdomain, n, will contain the points given by the indices

$$i = i' + \sum_{n'=1}^{n-1} d_{n'} \text{ for } i' = 1, \ldots, d_n \,,$$

where the valid range of indices is 1 to c, and any "ghost" cells which may be needed for the numerical method employed have not been included.

Since the decomposition is logically regular cartesian, the two-dimensional decomposition can be reduced to two independent one-dimensional decompositions using the above procedure.

Only after the subdomain bounds are calculated are the locations of land cells taken into account. Those subdomains with no ocean cells are dropped, reducing the total number from that requested before the decomposition. The decomposition is not recalculated when subdomains are dropped for this reason. Thus it is possible that when a decomposition into m by n subdomains is requested, the resulting number of subdomains will be less than $m \times n$.

Note that using the above algorithm to decompose a domain with $c^{(x)}$ by $c^{(y)}$ cells into $s^{(x)}$ by $s^{(y)}$ subdomains, it is possible for one subdomain to have as much as $\left(c^{(x)} \div s^{(x)}\right) + \left(c^{(y)} \div s^{(y)}\right) + 1$ more cells than another. This is unavoidable with this method and will lead to some level of load imbalance in the model.

Chapter 3

The Reduced Grid

In the preceding chapter, the details of the solution of the primitive equations on the standard latitude-longitude grid have been given. In this chapter, the modifications necessary for the implementation of the reduced grid will be explained. First some basic properties of the reduced grid are defined. Then the strategy for implementing this method into the LLNL ocean model is explained. Next some commonly used operations which act between the various resolution levels are introduced before moving to the algorithmic modifications to the discrete solution method. Finally the changes required for and limitations imposed by the method on parallel distributed memory computer architectures are given.

3.1 Basic Properties

This section will cover some basic properties of the reduced grid. First the effects of the staggered arrangement of variables on the ratio of resolution reduction between grid regions is presented. Then definitions and properties of global reduced grids are given for

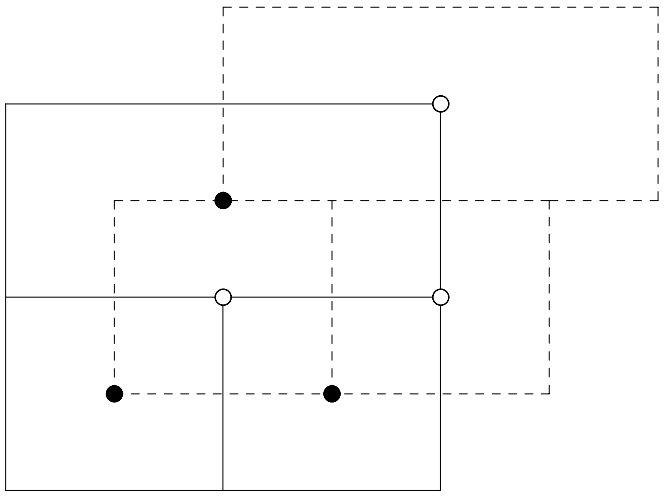

Figure 3.1: Tracer (solid) and velocity (dashed) boxes for an interface with a refinement ratio of two. Solid circles are tracer points and open circles are velocity points.

two specific cases. The effects of the staggered arrangement of variables on the locations of the interfaces are then shown, and two types of reduced grid interface are defined. Finally some details of indexing the reduced grid cells in longitude with a staggered mesh are presented.

3.1.1 Choosing the Refinement Ratio

In choosing the ratio of the number of grid cells on adjacent regions of differing resolution, a distinction between odd and even refinement ratios should be noted. In Figure 3.1, an interface is shown with a refinement ratio of two. On the tracer grid, cell boundaries on the coarser side of the interface align with cell boundaries on the finer side of the interface. However, for velocity grid cells, cell boundaries do not align, leaving fine cells adjacent to more than one coarse cell. Fluxes calculated across the interface from the fine cell would be divided into more than one coarse cell, adding complications. Also, the lack of consistency between the velocity and tracer interfaces this would impart is

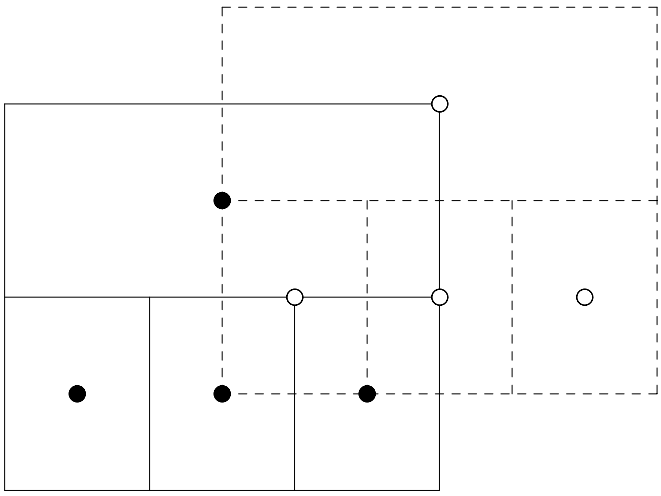

Figure 3.2: Tracer (solid) and velocity (dashed) boxes for an interface with a refinement ratio of three. Solid circles are tracer points and open circles are velocity points.

undesirable.

On the other hand, Figure 3.2 shows the relationship with a refinement ratio of three. For both the tracer and velocity cells, boundaries on the coarse side of the interface align with boundaries on the fine side. This simplifies the calculation of fluxes across the interfaces and allows a consistent set of operations for both the tracer and velocity grid variables. This argues for the use of only odd-integer refinement ratios. Also, since the goal of reducing the resolution toward the poles is to keep the grid cell size as constant as possible, lower ratios will achieve this better. Thus a refinement ratio of three will be used exclusively in this work.

3.1.2 Grid Definition

The ocean model uses a grid for which the tracer grid quantities have cell boundaries located at the equator, while the velocity grid quantities have cell centers at the equator. In the south, the presence of Antarctica means the grid can be terminated at 80 degrees

south, requiring no special concern with the pole. In the north, the singularity is removed by placing land cells at the highest row. While a velocity point would have been placed at the pole, this makes both the highest and second highest latitude velocity points zero. The highest latitude tracer cell is land, with a cell boundary at the pole.

The LLNL ocean code has the capability to use a grid in which the longitudinal grid spacing varies as a function of longitude, giving a "stretched" grid. Since this is rarely used for global grids, for simplicity in developing the reduced grid code, this capability is eliminated. Also, there is the capability to use a grid in which the latitudinal grid spacing varies as a function of latitude. While this option is not used in the model runs presented here, this capability has been retained with the addition of the reduced grid.

The optimal locations for the changes in resolution must be determined with the goal of keeping grid cells as uniform in size as possible. With a three-to-one ratio of longitudinal resolution between adjacent grid regions, the cell size will nearly triple when moving across an interface toward the poles. Choose a reference grid spacing, Δx_{ref}, no smaller than two-thirds the size of the equatorial grid spacing, Δx_{eq}. All grid cells will remain within fifty percent of the size of Δx_{ref} if the grid is coarsened at each latitude at which the cells would be smaller than half the reference size without coarsening. Two such choices of Δx_{ref} are presented here: Δx_{eq} and $(2/3)\Delta x_{\mathrm{eq}}$.

For the following grid definitions, the latitudinal spacing is chosen to be 2.5 degrees, as this will be the spacing for the global model runs which will be presented later.

The choice of reference grid spacing of $(2/3)\Delta x_{\mathrm{eq}}$ results in a grid with properties shown in Figure 3.3. Tracer grid interfaces are located at 70 and 82.5 degrees. This grid moves the changes in resolution as far poleward as possible while still keeping cells as uniform in size as possible. The number of grid cells is reduced from $36 \times N_{\mathrm{lon}}$ per

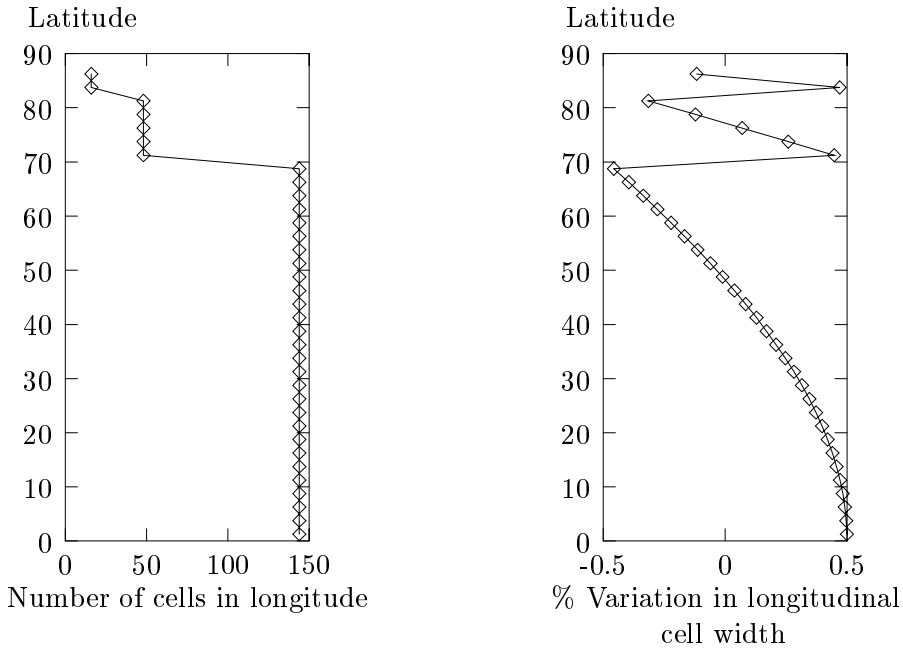

Figure 3.3: Reduced grid properties for $\Delta x_{\text{ref}} = (2/3)\Delta x_{\text{eq}}$

hemisphere to $(28+5/3+3/9) \times N_{\text{lon}}$, where N_{lon} is the number of cells in the longitudinal direction at the equator. This is a $16\frac{2}{3}$ percent reduction in the number of cells.

The choice of reference grid spacing of Δ_{eq} results in a grid with properties shown in Figure 3.4. Tracer grid interfaces are located at 60 and 80 degrees. This grid keeps the smallest grid cell as large as possible while also still keeping cells as uniform in size as possible. The number of grid cells is reduced even more, to $(24 + 8/3 + 4/9) \times N_{\text{lon}}$ per hemisphere. This is a reduction in cells of nearly 24.7 percent.

The latter grid will be used for the global runs in this work for several reasons. First, the larger size of the smallest grid cells should allow for the largest increase in timestep. Secondly, the number of cells is reduced the most, which should give the largest reduction in execution time per step, as well as present the greatest challenge for load balancing the parallel runs. Lastly, the interfaces are located at lower latitudes, so their influence

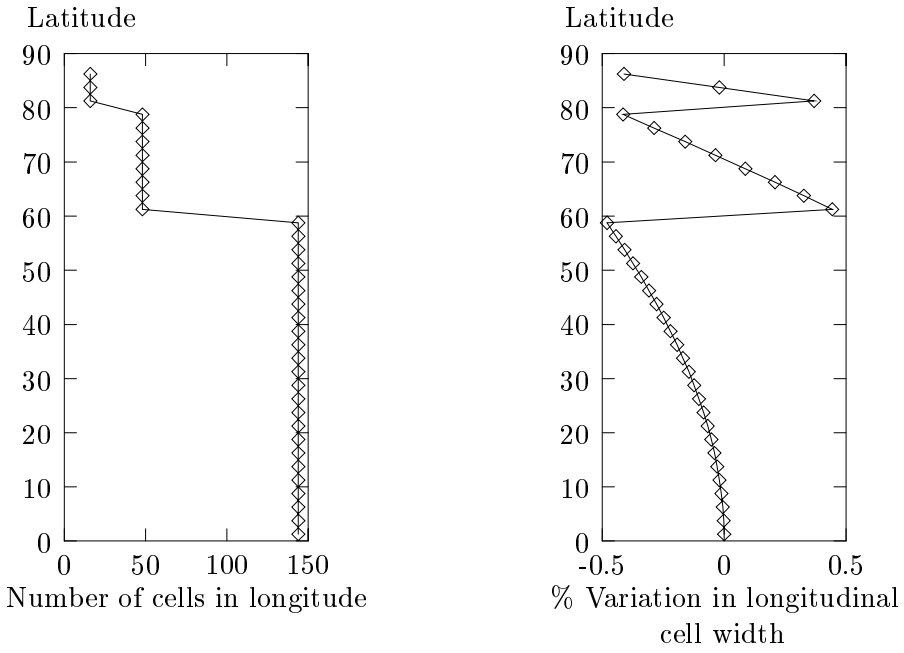

Figure 3.4: Reduced grid properties for $\Delta x_{\text{ref}} = \Delta x_{\text{eq}}$

should be more strongly felt throughout the model solution. This is beneficial because the goal here is to investigate what these effects are — not necessarily to reduce them to the smallest levels possible.

3.1.3 Interface Location and the Staggered Grid

The staggered placement of the variables on the grid in the horizontal was described in Section 2.3.1. In the above definition of the reduced grid, the location of the tracer grid interfaces was specified, but because of the staggered arrangement, that alone does not specify the location of the velocity grid interfaces.

Actually, there are two options for the relative locations of the tracer and velocity grid interfaces. Figure 3.5 shows the option in which the velocity grid interface is located toward the coarser grid relative to the tracer interface. In this situation, each tracer grid

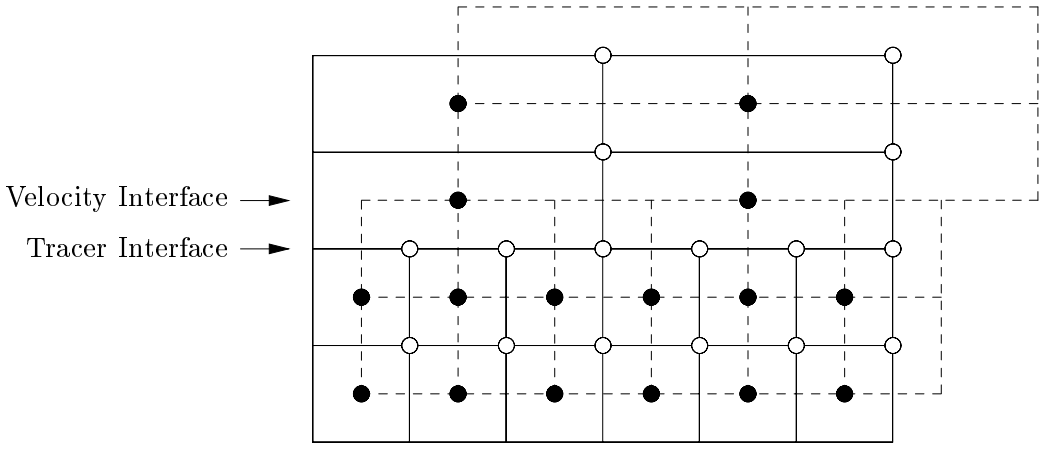

Figure 3.5: A "type-1" interface showing the relative locations of the tracer grid (solid circles and lines) as well as the velocity grid (open circles and dashed lines).

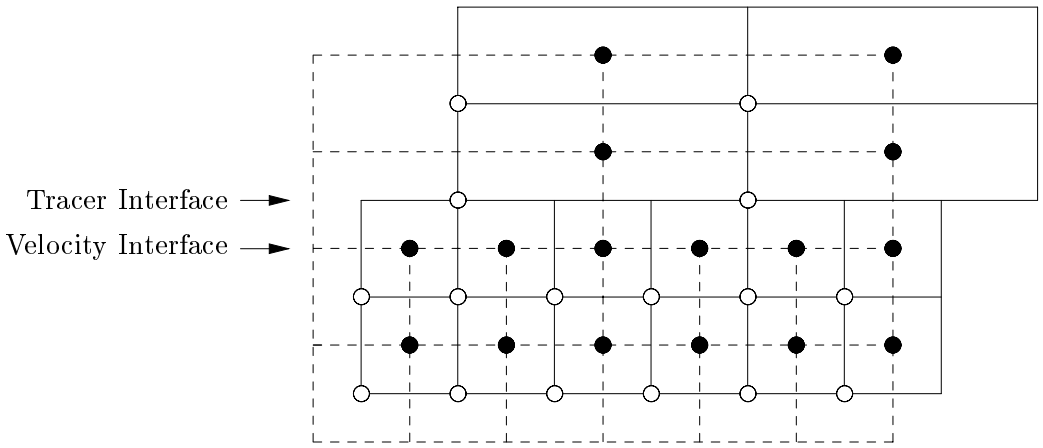

Figure 3.6: Same as Figure 3.5 except for a type-2 interface.

cell on the finer side of an interface has a velocity grid point at each corner. In the other option, shown in Figure 3.6, the velocity grid interface is located toward the finer grid relative to the tracer grid interface. Here it is the velocity grid cell on the finer side of the interface which has a tracer grid at each corner.

Another way to define the two cases is to state which grid — coarse or fine — the velocity points lie on at the tracer grid interface. In the first example, the velocity points at the tracer grid interface are fine grid points. In the other, they are coarse grid points. Here the two types will be referred to as "type-1" and "type-2" interfaces, respectively.

The choice of interface type has a bearing on the details of the finite difference calculations at the interfaces. For example, with a type-1 interface, tracer advection requires interpolation of the tracer quantity only. However, with a type-2 interface, tracer advection requires interpolation of both the tracer quantity and the velocities at the interface. Another difference is that the quantities involved in gradient terms along the interfaces — pressure and surface height — are tracer grid quantities. Thus, a type-1 interface will interpolate a quantity and subsequently take a derivative in the direction of the gradient. Note that this is never needed with a type-2 interface. This could be a potential source of numerical noise in the calculation, and more will be discussed later on this subject.

Both types of interfaces will be implemented.

3.1.4 Longitudinal Indexing and the Staggered Grid

Since there will be interaction between grids of differing resolution across the interfaces, it will be important to identify the relationships between cells on each side of the interfaces. Here the staggered relationship of the model variables also has an effect, leading to different treatments for tracer and velocity grid variables.

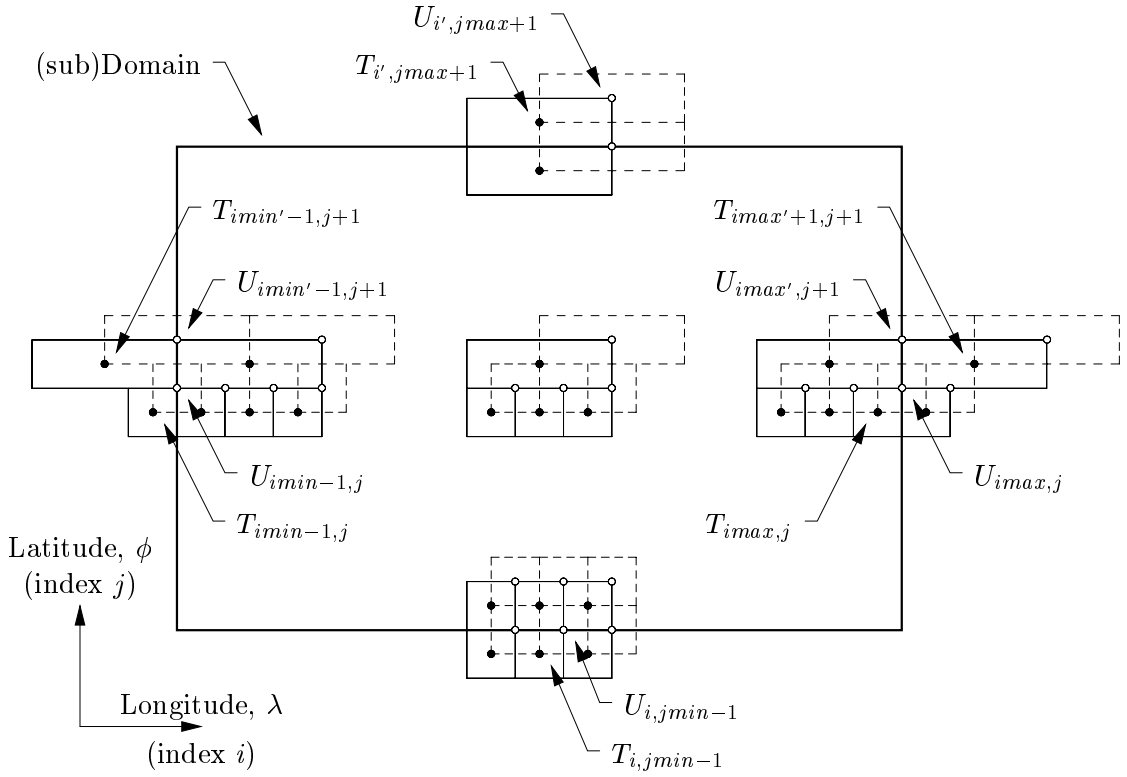

Figure 3.7: Grid overview

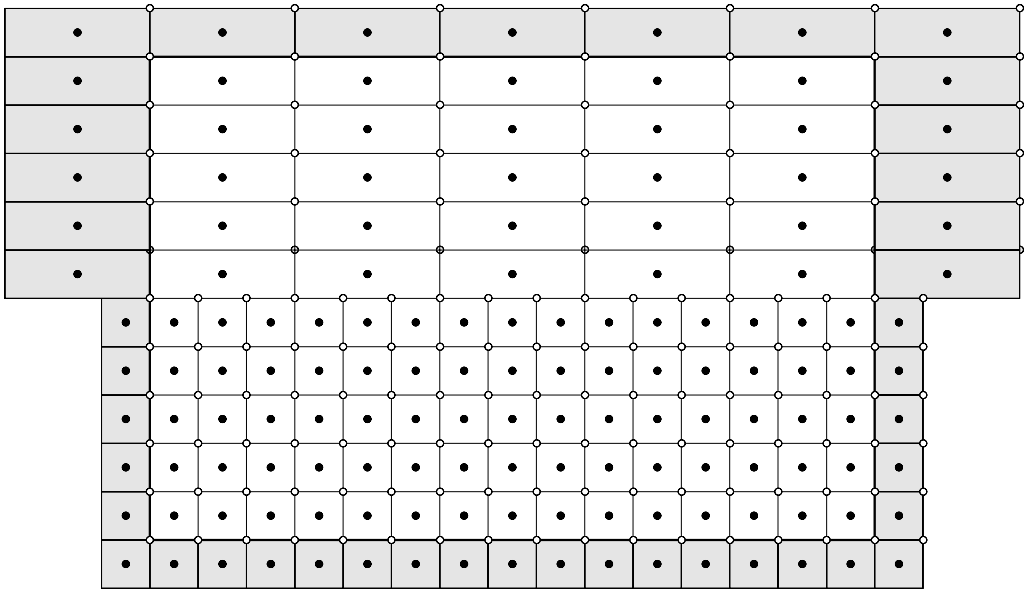

Figure 3.8: Tracer grid (sub)domain

For reference, an overview diagram of a domain (or subdomain) containing one type-1 interface is shown in Figure 3.7. Note the influence of the relative grid staggering at the domain boundaries and at the interface in particular. On each subgrid, the relative locations of the boundary grid cells and the domain border (shown as a heavy line in the figure) are as they would be on a regular grid of that resolution. However, at the interface, the stagger of the tracer and velocity grids leads to a different relationship across the interface for the tracer grid than the velocity grid.

Figure 3.8 shows the same type of grid as Figure 3.7 but only the tracer grid part of it. Within the subdomain, for each coarse cell at the interface there are three corresponding fine cells across the interface. Outside the subdomain, there is one "ghost" cell, shown shaded in the figure, on all sides.

Looking at Figure 3.9, which shows the velocity grid part, it is immediately apparent that the velocity grid has different characteristics. On the western and southern edges

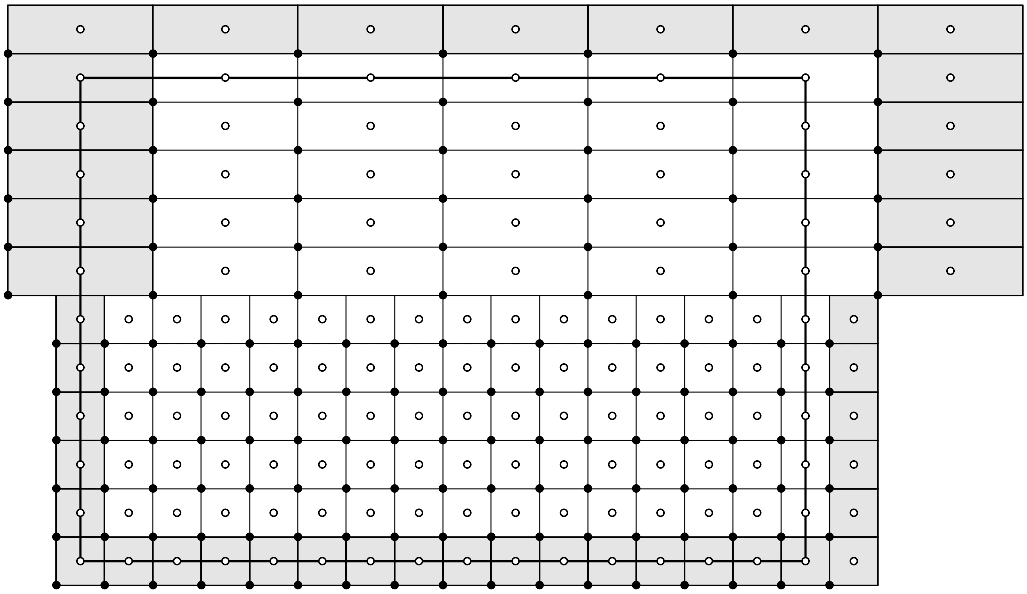

Figure 3.9: Velocity grid (sub)domain

of the subdomain, the "ghost" cells lie on the subdomain boundary. On the eastern and northern edges, they lie outside the subdomain boundary, with the next most westerly (southerly) cell lying on the boundary. This yields a different relationship across the interface at the domain boundaries for the velocity grid than the tracer grid. At the westernmost coarse cell, there are two corresponding cells across the interface, one being a "ghost" cell. At the easternmost coarse cell, there are no cells across the interface. The easternmost fine "ghost" cell corresponds to the second most eastern coarse cell.

This difference between the tracer and velocity grids at the interface must be kept in mind as the reduced grid algorithms are presented later. Affected procedures include the filling, copying, averaging, and interpolation of Section 3.3 and the communication necessary for parallel implementation.

3.2 Implementation Strategy

Since the reduced grid will be used in the framework of an existing, well-developed model (see intro to Chapter 2), the implications of any significant code changes must be carefully considered. If the changes made are large enough to require alterations of the basic design strategies of the original model, the resulting product will no longer be merely a modification, but a whole new model. This is not desired for two main reasons. First, comparisons of the original and reduced grid models should ideally reflect only the change in numerical grid, not changes in code layout, data restructuring, or parallelization strategy. While such changes may been improvements in some respects, they would only obscure the true goal of this project. Second, the less obtrusive the required changes can be made in this particular implementation, the more likely the results may be of interest to others with similar models.

Thus the general design strategy should preserve the overall layout of the model. When practical, there should be no modification to the outer loops or control structures, as well as to the overall layout of the data structures. Benefits to this approach include easier incorporation of parallel processing techniques already built into the code and the preservation of the method of coupling the OGCM to other climate models.

This section will describe in some generality the main methods used to achieve this desired result. First, the minimum modifications required to the data structures in order to keep the algorithms intact are described. Then those modifications which reduce the computation to only those grid cells necessary for our reduced grid are outlined. Finally, it may be of interest to outline a different implementation strategy for comparison, and one such method will be described at the end of the section.

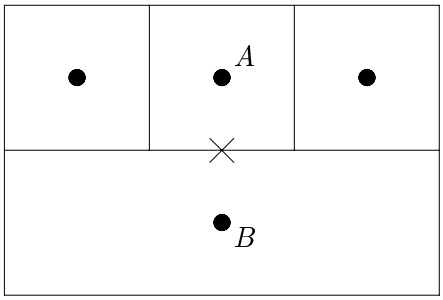

Figure 3.10: An interface with no dummy row

3.2.1 Dummy Interface Rows

In pursuing the preservation of the model data structures, balance must be struck with the goal of minimizing the changes required to the model algorithms. Therefore, the one such modification made is the addition of extra storage at the location of each interface in model arrays with latitude as a dimension. For example, a two-dimensional array over latitude and longitude would gain three new rows for the grid described in Section 3.1.2. Section 3.4 will show how this modification will allow minimum modification to the model algorithms.

The standard numerics for the MOM type models require data from each cell's nearest neighbors and no further. However, if a data cell is located adjacent to a grid interface, its neighbor in the direction of the interface may not be addressable by a simple increment or decrement of an index, and there may be more than one neighboring cell across the interface. Because of this, the standard finite-difference numerics or algorithm would have to be modified at grid interfaces. Consider the simple case illustrated in Figure 3.10. Here both cell A and cell B need to use a flux calculated on their shared face. However, due to the difference in resolution, the fluxes for the two cells will be related, but different in general. If this were also the layout of the data in memory, there would only be one

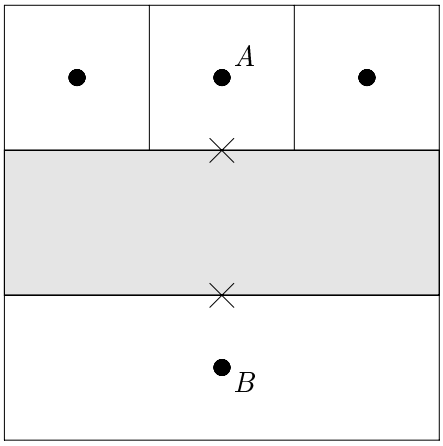

Figure 3.11: An Interface with a dummy row

location in which to store two different pieces of data. Thus the standard algorithm for computing the rate of change of a cell's value would have to be modified to treat special cases.

However, by the insertion of an extra row between the two grids into the data structures at the interface, as illustrated in Figure 3.11, both cells have their own data for the flux, and the algorithms for each cell will not need to be changed. Of course, the cells in the added row are a purely algorthmic device and do not correspond to physical locations in the domain. Therefore, most cell computations are able to skip these interface rows.

Now in order to carry out the standard computations for cells bordering an interface, various interpolations and averages and the like will be done at the interface and in the interface dummy row. Therefore, a way of indexing, or referring to the latitude index of, an interface row is required. Unfortunately, this task is complicated by both the relative staggering of the tracer grid and the velocity grid, as well as the possible orientation of the interface in either of two directions. By the latter, I mean that we may have an interface where a coarser grid is either to the north of a finer grid or to the south. I will refer to

these orientations as "north" and "south", respectively.

Defining the index of an interface dummy row on the tracer grid by the index j, Figure 3.12 shows the relative indexing of the velocity grid for both north and south facing interfaces of type-1 (see Section 3.1.3 for definitions of interface type.) Thus, on the tracer grid, for a north facing interface, the coarse grid is at an index one greater than the interface row, while the fine grid is at and index one less. The reverse is true for south facing interfaces. The same relationship holds for the tracer grid at a type-2 interface, shown in Figure 3.13.

On the velocity grid, however, the staggering of the grids changes this relative indexing. With the tracer grid interface index still defined as j, the index of the interface dummy row on the velocity grid is given by j for a north facing type-1 interface, but $j-1$ for a south facing type-1 interface. The coarse and fine sides are again one higher or lower than that, depending on the orientation. For a type-2 interface, the velocity grid interface dummy row is given by $j-1$ for a north facing interface, but j for a south facing interface.

3.2.2 Loop Modification

The data structures remaining after the addition of interface rows as shown above does not eliminate the extra memory locations which are no longer used due to the coarsening of the grid. However, if calculations were still performed on these extra grid cells, some of the benefit of the reduced grid method would be lost. Figure 3.14 shows an example of what a two-dimensional array might look like in a reduced grid scenario as implemented here. The shaded cells are not used by the code, so a simple method needs to be devised to avoid performing calculations in these cells. Simplicity is needed to remain faithful to the idea of altering the code as little as possible.

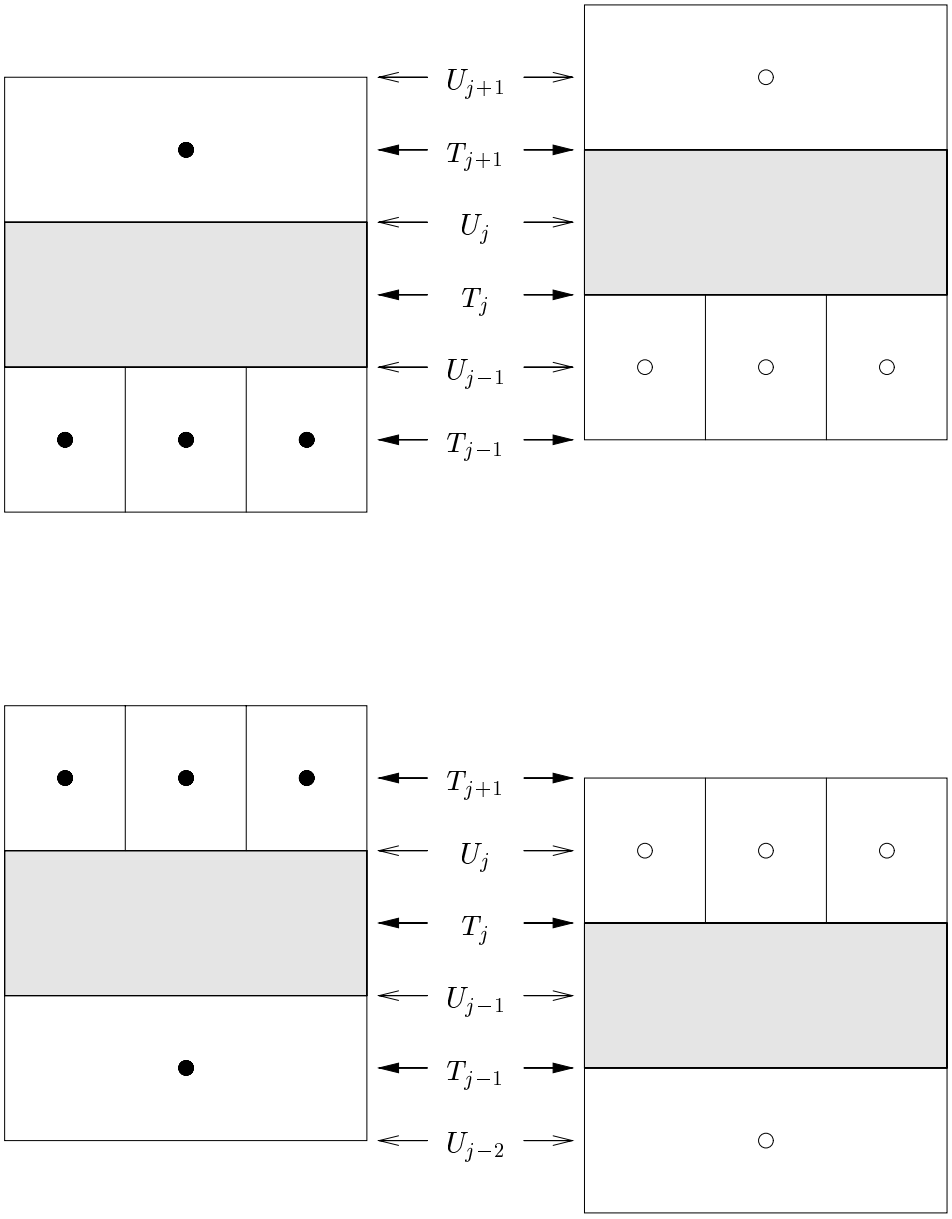

Figure 3.12: Type-1 north and south facing interface latitudinal indexing. Dark circles are tracer grid points and open circles are velocity grid points.

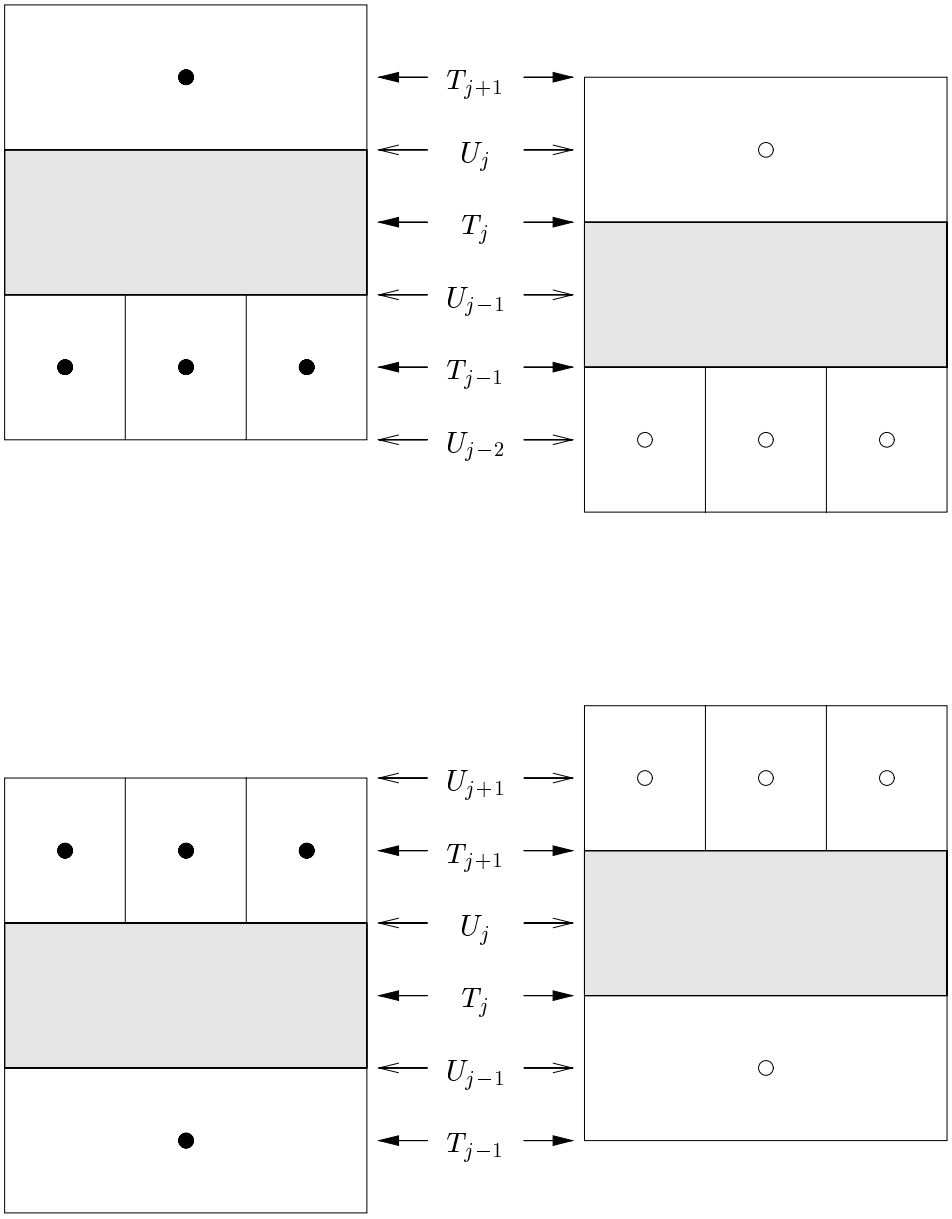

Figure 3.13: Type-2 north and south facing interface latitudinal indexing. Dark circles are tracer grid points and open circles are velocity grid points.

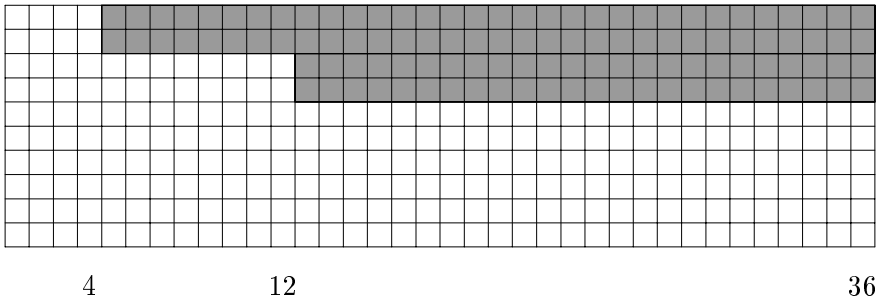

Figure 3.14: An example two-dimensional, reduced grid array

The following FORTRAN pseudocode illustrates the standard convention for looping over a data structure followed by the method chosen here to perform the same loop over a reduced grid in a way that bypasses the unused elements.

```
do j = jmin, jmax
  do i = imin, imax
    array(i,j) = x * y   ! Some calculation
  enddo
enddo

do j = jmin, jmax
  do i = imin_new(j), imax_new(j)
    array(i,j) = x * y   ! Some calculation
  enddo
enddo
```

The vectors `imin_new` and `imax_new` are defined in such a way that interface rows are skipped and unused array elements are skipped in regions of coarsened resolution. This is accomplished by setting `imax_new` to a value less than `imin_new` for values of `j` corresponding to interface dummy rows. Also, the value of `imax_new` will vary from `imax` in regions of standard resolution, to smaller and smaller values as the grid is coarsened.

Both the stagger of the grids, discussed in Section 3.2.1, and the two different types of interfaces, discussed in Section 3.1.3, add complications which will require four separate versions of the `imin_new` and `imax_new` vectors. One set, herein named `imin_t` and

`imax_t`, will skip tracer grid interface rows and extra tracer cells. Another, named `imin_u` and `imax_u`, will skip velocity grid interface rows.

The third and fourth sets are used only for the type-1 and type-2 interfaces, respectively. For a type-1 interface, `imin_s` and `imax_s` will skip the coarse tracer row adjacent to an interface, but for the tracer interface dummy row will be defined the same as the fine row adjacent to the interface. For a type-2 interface, `imin_r` and `imax_r` will skip the coarse velocity row adjacent to an interface, but for the velocity interface dummy row will be defined the same as the fine row adjacent to the interface. The use of these index arrays will be described in Section 3.4.

Some definitions will ease the use of these arrays and loops when describing operations at a grid interface. The tracer interface dummy row will be referred to as the "tracer interface row". The coarse tracer grid row adjacent to an interface will be referred to as the "coarse tracer row" or "coarse row" if the context makes it clear to which grid it is referring. The fine tracer grid row adjacent to an interface will be referred to as the "fine tracer row" or "fine row". The same type of terminology will be used for the velocity grid.

A loop over latitude and longitude as above using the `imin_t` and `imax_t` indices will be called a "loop over tracer indices" or a "T-loop" for short. A similar loop using `imin_u` and `imax_u` will be called a "loop over velocity indices" or "U-loop." Likewise, with `imin_s` and `imax_s` or `imin_r` and `imax_r` it will be called a "loop over special indices" or "S-loop." The rows thus skipped are referred to as the tracer interface rows, velocity interface rows, and special interface rows, respectively. The first two types of skipped rows correspond to the added dummy rows of Section 3.2.1, while the third type is one of the rows adjacent to an interface dummy row.

3.2.3 An Alternate Strategy

The above description of implementation strategy has been guided by the desire to keep the overall model structure and logic intact. However, it is useful to consider situations in which this is unimportant in comparison to other needs or in which the model uses a completely different code structure. In either of these situations, the following strategy may be of interest.

A grid of the type shown in Figure 1.1 may be thought of as one grid which is coarsened toward the poles, as has been done throughout this work. However, it may equivalently be considered multiple grids which are coupled together at their boundaries. Whereas the LLNL ocean model is designed and written to operate on one grid covering the whole model domain, the same algorithms could be rewritten to operate on an arbitrary number of grid sections with boundary conditions specified for each. This is obviously a more flexible design, as the case of one global grid with periodic boundaries is simply one configuration it could handle.

The reduced grid could be implemented in this model design by specifying the grid sections corresponding to each latitudinal band of constant resolution. The operations necessary to couple the differing resolutions would then be used to calculate the boundary conditions at the northern and southern boundaries of the individual grids. Each individual grid would contain (at least) one row of "ghost" or "dummy" cells at the boundaries, which would obviate the need for adding interface rows.

This strategy is not without its difficulties. For example, parallelization of this type of model is less straitforward. Since the code would iterate over each model grid, global decomposition must give way to decomposition of subgrids. Each grid could then be

domain decomposed and thus parallelized, but if some grids are very small this could reduce the overall parallel efficiency. Other parallel methods could be applied to overcome these difficulties, but they would necessarily add complications. As another example, the reduced grid algorithms require fluxes to match across interfaces. This requires a particular sequence of calculation, with added calculations at the interface occuring in the middle of a timestep. These steps would further constrain the iteration of such a model over subgrids.

This example is given not to show the superiority of the implementation strategy chosen, but merely to give an alternate implementation with which to compare. All of the difficulties are undoubtedly surmountable, and likely have been dealt with in other models. Here it is noted that rewriting a model in this manner would be a sizeable project, with many details to work out in order to be successful. This consideration combined with the desired model features given at the beginning of the section are enough to leave this type of implementation for future work.

3.3 Interface Operations

Certain operations on data near grid interfaces will recur often enough in the numerical algorithms to devote a section to describing them in generality. In the code implementation, these operations are written in a very flexible manner, and their descriptions here will reflect that flexibility. The details of how and where these operations are used will be given in the section following which describes the actual algorithmic changes.

Another method of referring to the locations at which quantities are defined will be used at times. For example, a flux which is defined on the north face of a tracer cell can

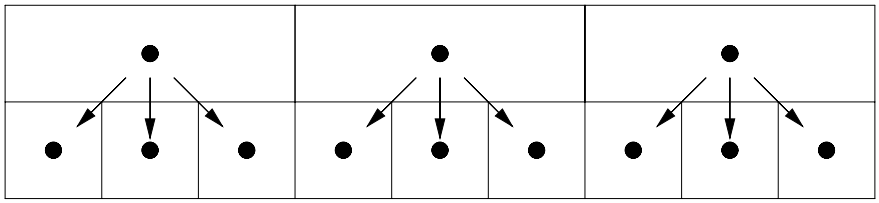

Figure 3.15: Point-to-point fill

actually be better described as being a tracer grid quantity longitudinally and a velocity grid quantity latitudinally because it is aligned with the tracer points in longitude and the velocity points in latitude. There are many other such quantities like this, and the distinction is particularly important when discussing the following interface operations. Therefore, a definition of terms is in order. A quantity is either T_λ or U_λ depending on whether it is aligned with the tracer or velocity grid points in longitude. Each quantity is also either T_ϕ or U_ϕ depending on whether it is aligned with the tracer or velocity grid points in latitude.

As discussed in Section 3.1.4, there is a difference in how tracer and velocity grid longitudinal indices align with their neighbors across interfaces. Therefore each of the operations below will have one version for T_λ quantities and one for U_λ quantities. Taking the example operation of averaging, the former version can be referred to as an "average over tracer indices" or a "T-type average" for short. The latter version can be referred to as an "average over velocity indices" or "U-type average."

3.3.1 Point-to-Point Fill

This is a coarse-to-fine operation between grids near an interface, illustrated schematically in Figure 3.15. The value of each coarse grid cell is copied into all of the fine cells to

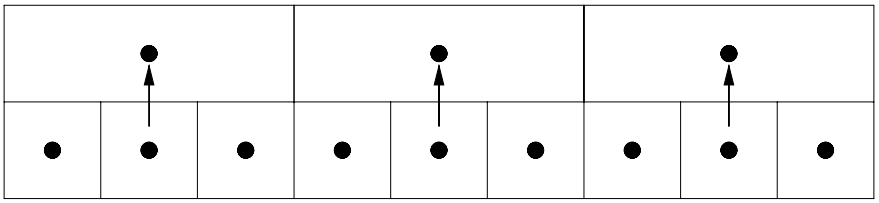

Figure 3.16: Point-to-point copy

which it corresponds. For this operation on the tracer grid, the coarse ghost cells at the eastern and western boundaries correspond to the fine ghost cells. For the velocity grid, the coarse ghost cell at the western boundary corresponds to the two westernmost cells of the fine grid. See Section 3.1.4 for details. If the ghost cells of the coarse grid are set properly before the fill, no communication is necessary between subdomains.

This operation is analogous to a zeroth-order interpolation. It is used in this capacity and for various initialization and input/output operations.

3.3.2 Point-to-Point Copy

This is a fine-to-coarse operation between grids near an iterface, illustrated schematically in Figure 3.16, used prior to the calculation of certain zonal fluxes for which the shared points at the interface are required in both the fine and coarse grid calculations. Those fine resolution points which correspond exactly in longitude to coarse resolution points are copied to the corresponding coarse cells. Those fine resolution points which don't correspond in longitude to coarse resolution points are unused in this operation. For the tracer grid, the eastern and western coarse ghost cells are not defined in this operation, as they do not have directly corresponding fine cells. On the velocity grid, the eastern ghost cell likewise does not have a corresponding fine cell and is therefore undefined for

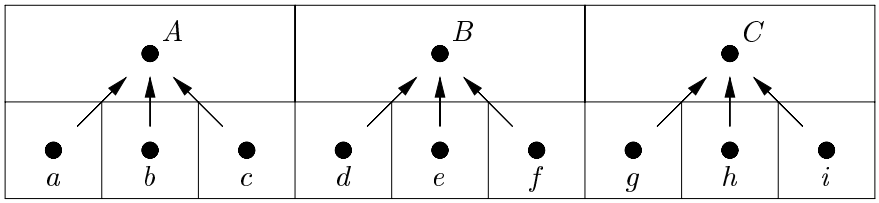

Figure 3.17: Averaging

this operation. However, the western velocity ghost cell does correspond directly to the fine ghost cell. Again, see Section 3.1.4 for more details. If the coarse ghost cell values are required after this operation, communication between subdomains will be required for the eastern and western tracer ghost cells and the eastern tracer ghost cell.

3.3.3 Average

This is a fine-to-coarse operation, illustrated schematically in Figure 3.17. The value of each fine grid cell is averaged together with all other (three total) fine cells that correspond to the same coarse grid cell. The resulting average is copied to the corresponding coarse cell. Depending on the quantity being averaged, this operation may in fact be a summation. All of the ghost cell values are undefined for this operation. Thus, if they are needed after the average operation, communication between subdomains will be required.

In the presence of topography one or two of the quantities being averaged may be zero while the others are nonzero. This would occur when the finer row adjacent to an interface has both land and ocean cells, while the coarser row is an ocean cell. For example, if in Figure 3.17, points B, e, and f were ocean points, while point d was a land point. When these quantities are fluxes, the zero values should just be averaged along with the others, as the resulting flux will be for the entire length of the fine-coarse shared boundary. That

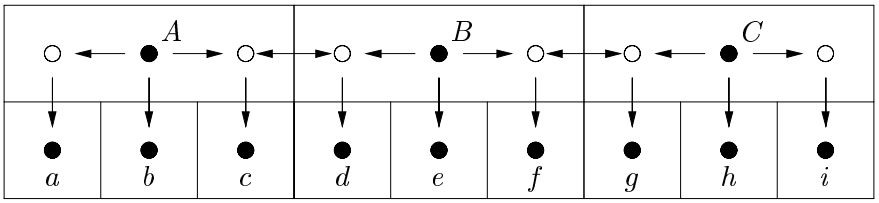

Figure 3.18: Interpolation

is, conservation of fluxes at the interface requires

$$F_c \cdot h_c = \sum_{n=1}^{3} F_n \cdot h_f, \tag{3.1}$$

where F_c is the coarse cell interface flux, F_n is the nth fine cell interface flux, and h_c and h_f are the coarse and fine cell interface edge lengths, respectively. Since longitudinal mesh spacing is not allowed to vary as a function of longitude,

$$h_c = 3h_f, \tag{3.2}$$

giving

$$F_c = \frac{1}{3} \sum_{n=1}^{3} F_n, \tag{3.3}$$

which is the simple averaging operation. So conservation requires the inclusion of zero fluxes due to land cells in the averaging operation.

This operation is used for all fluxes which cross the interfaces. For example, the tracer advective fluxes from the fine grid row adjacent to an interface are averaged and used as the flux to the coarse grid row adjacent to the interface.

3.3.4 Interpolation

This is a coarse-to-fine operation, illustrated schematically in Figure 3.18. It is the most complex of the operations in this section, as there are a number of options. The

values of the fine grid are determined from the coarse grid by polynomial interpolation, cubic spline interpolation, or a point-to-point fill as noted above. Numerical details of interpolation techniques may be found in Section A.4 and [55]. Polynomial interpolations from linear to cubic are implemented using the nearest cell values available. For example, cubic polynomial interpolation will always use two cells to the left and two to the right of the point at which the value is needed. Linear interpolation will always use the nearest cell to the left and the right.

The boundary conditions for interpolation at land cells are no-flow for the velocity and no-flux for tracers. That is, for purposes of interpolation, velocity values at land cells are set to zero, while tracer values at land cells are set to the value of the cell immediately adjacent to the land cell. So, if point A in Figure 3.18 corresponds to a velocity grid land point, it will be used in an interpolation with a value of zero. However, if point A corresponds to a tracer grid land point, it will be used in an interpolation with the value of point B, assuming point B is not also a land point.

Topography has further influence on the interpolation in that it can require the order of the interpolation to be reduced when quadratic or cubic polynomial interpolation is used. For example, when interpolating to a point d in Figure 3.18, where points A and B are land and ocean points, respectively, cubic interpolation will be reduced to quadratic interpolation. If point C is also a land point, then the interpolation is further reduced to linear.

Because of the difference in the definition of a tracer land cell and a velocity land cell discussed in Section 2.3.2, there is yet another difference between tracer and velocity grid interpolation near topography. Referring to Figure 3.19, point A is a tracer land cell. Therefore interpolation point B, located within the tracer cell defined by A, is also a land

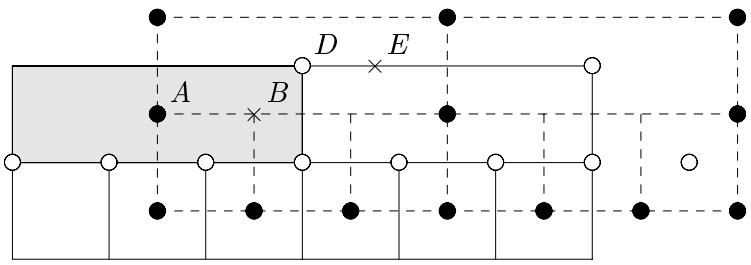

Figure 3.19: Interpolation near topography

cell. Point D is a velocity grid point that is a land point. However, even though point E is with the velocity cell defined by D, it is not located on a material boundary and is not itself a land point. Interpolation must be used to determine a value at E.

A further complication of the presence of topography is its effect on cubic spline interpolation. Polynomial interpolation can be calculated as a weighted average of the interpolation points, with the weights calculated once at initialization. Cubic spline interpolation, however, requires a linear solve which is dependent on all the values along the line of interpolation. Since complicated topography may break latitude lines into many pieces, this will in general require many independent interpolations. This process is made more difficult by the need to calculate these independent interpolations between land cells on latitude lines within the framework of domain decomposition.

Subdomain communication requirements are dependent upon the type of interpolation. The point-to-point fill and linear types of interpolation require communication only to set the initial coarse ghost cell values. Quadratic and cubic polynomial interpolation require an extra ghost cell to be communicated on each end of the subdomain. Cubic spline interpolation is non-local, requiring a solve across the entire latitude row even when no topography is present. This involves much greater communication expense than the

polynomial interpolations. Combined with the difficulties in dealing with topography, cubic spline interpolation is used only in the simpler test cases of Chapter 4. Further work in using cubic splines as the basis of interpolation could follow the example of efficient filtering techniques for parallel processors in [46].

3.3.5 Reduction

This operation is a fine-to-coarse operations analogous to the point-to-point copy. However, a reduction does not just affect one coarse and one fine row near an interface. Instead it affects every row, changing each row from the standard grid — that is, not a reduced grid — resolution to its reduced grid resolution. Thus, taking a field defined on a standard grid and applying a reduction operation yields a field defined on a reduced grid. However, like a point-to-point copy and unlike an average, only the standard resolution points corresponding directly to coarsened resolution points are involved in the reduction. This operation is used in initialization and for input/output routines.

3.3.6 Expansion

This operation is a coarse-to-fine operation analogous to the point-to-point fill. However, like the reduction, it does not just affect one coarse and one fine row near an interface, but it affects every row, changing each row from the reduced grid resolution to the standard grid resolution. In some sense, it is the opposite of a reduction operation, though a reduction is not truly reversible. That is, a reduction takes some multiple of three points and converts them to just one point, possibly losing some information in the process. The expansion takes one point and converts it to some multiple of three points. This operation is used for input/output routines.

3.4 Modifications to the Discrete Equations

The changes necessary for the solution on the discretized equations of Section 2.3 on a reduced grid are presented in this section. Most of the changes are really changes in the procedure of the solution, rather than changes to the fundamental numerics. Because of this, most of the quantities defined in Section 2.3 can be directly used in this section. While the descriptions may seem to indicate large code modifications, the underlying operations taking place and modification to the code loops are rather simple. The complexity arises mostly from the grid stagger and the two possible types of grid interface (see Section 3.1.3). Therefore, summary comparisons of the standard and reduced grid procedures will be given for the more complex parts. In this section, the changes to the initialization and surface boundary conditions will be presented first. Then, each piece of the discretized primitive equations will be dealt with separately, as was done in Section 2.3.

3.4.1 Initialization

Topography

Section 2.3.2 described the definition of topography on the standard grid. The modifications required for the reduced grid involve both the respecification of the topography data file and the reading of this data into the data arrays in the model. It is actually not required that the topography data file be modified at all. However, for direct comparison between the standard and the reduced grid, the topography used in the global runs described later is modified. The details can be found in the section describing the run parameters, Section 5.1.

With a reduced grid, a standard grid topography data file is read at model initial-

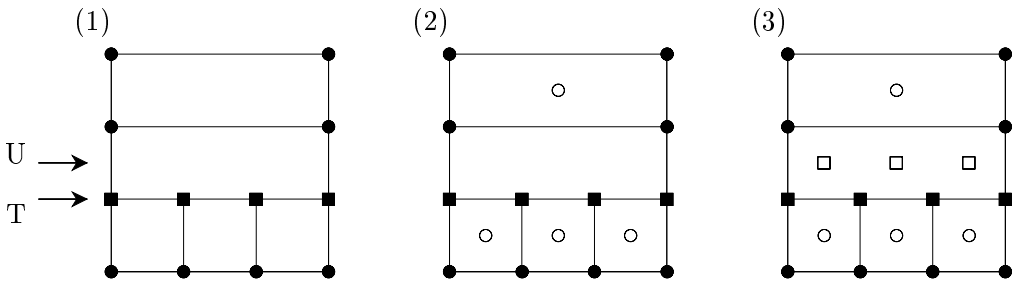

Figure 3.20: Procedure for setting velocity grid topography at a type-1 interface. Labeled arrows indicate interface rows. (1) Fill the tracer grid interface values (dark squares) from the coarse row topography data (dark circles). (2) Calculate velocity topography (open circles) as minimum of neighboring tracer topography, skipping velocity interface row. (3) U-type point-to-point fill (open squares) from the coarse velocity row to the interface row.

ization and then converted to the reduced grid resolution. First, dummy interface rows are added to the topography data, as described in Section 3.2.1. Then a T-type reduction operation is applied to the data, as described in Section 3.3. This gives an array with interface rows added and the correct resolution data, but the data in the interface rows is not yet set correctly.

The topography of an interface row is defined to be the same as the topography of the adjacent coarse row, but at the resolution of the adjacent fine row. This can be accomplished for the tracer grid by the point-to-point fill operation of Section 3.3, operating between the coarse row and the interface row.

For the velocity grid topography, which was defined to be the minimum of the four adjacent tracer grid depths, the procedure is slightly more complicated and is dependent on the type of the interface, as defined in Section 3.1.3. For the type-1 inteface, a U-loop is used, calculating the minimum of the four adjacent tracer grid depths as on the standard grid. Then a U-type point-to-point fill is used to set the velocity interface row from the coarse row. This is shown schematically in Figure 3.20. For the type-2 interface,

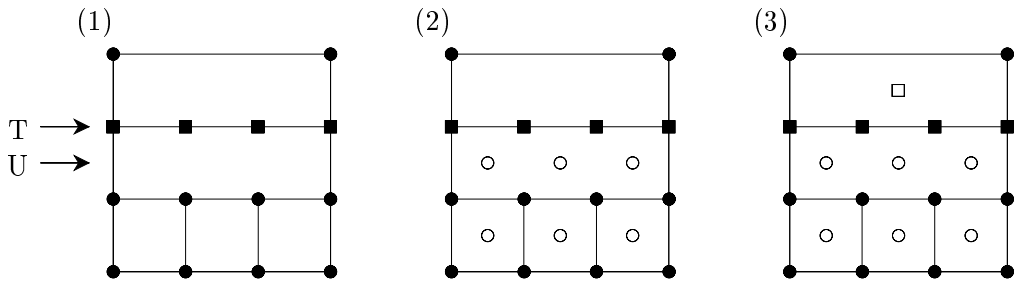

Figure 3.21: Procedure for setting velocity grid topography at a type-2 interface. Labeled arrows indicate interface rows. (1) Fill the tracer grid interface values (dark squares) from the coarse row topography data (dark circles). (2) Calculate velocity topography (open circles) as minimum of neighboring tracer topography, skipping coarse row. (3) U-type point-to-point copy (open squares) from the velocity interface row to coarse row.

an S-loop is used, calculating the minimum of the tracer grid depths as before. Then a U-type point-to-point copy is used to set the coarse velocity rows from the interface rows. This is shown schematically in Figure 3.21.

Surface Boundary Conditions

Data needed to calculate surface boundary conditions, consisting of surface wind stress, temperature, and salinity, can be read from data files. When this is the case, the data is treated much like the topography data. First, dummy interface rows are added to the data arrays. Then a reduction operation is applied, resulting in an array with added dummy rows and the correct resolution of the data. For the boundary condition arrays, the dummy interface values are never used, so they are left undefined.

3.4.2 Tracer

The discretized tracer transport equation, given by (2.94), consists of an advective and a diffusive term. The general procedure for advancing this equation on the standard grid compared to the procedure on a reduced grid as follows (column-only operations

omitted):

Standard Grid	Reduced grid
	• Interpolate tracers and velocities (if necessary) at interfaces.
• Calculate horizontal advective velocities.	• Calculate horizontal advective velocities.
	• Average meridional advective velocity at interfaces.
• Calculate vertical advective velocity.	• Calculate vertical advective velocity.
• Calculate advective and diffusive fluxes.	• Calculate advective and diffusive fluxes.
	• Average meridional advective and diffusive fluxes at interfaces.
• Calculate divergences of advective and diffusive fluxes.	• Calculate divergences of advective and diffusive fluxes.

By this comparison it can be seen that the required modifications to the calculation procedure consist mainly of extra steps inserted between standard calculations and modifications to loops as described in Section 3.2.2. With this overview in mind, the details are given below.

Advective Velocities

Following Section 2.3.5, the advective velocities of (2.95) through (2.97) are first calculated. The advective velocity on the eastern face of the cell, AdvVel_Te, is a T_ϕ quantity (see Section 3.3) and is calculated with a T-loop, which skips the tracer interface row. This quantity is not needed for the interface row, so it is left undefined there.

AdvVel_Tn is a T_λ /U_ϕ quantity and is treated differently for type-1 and type-2 interfaces. With a type-1 interface, it is first computed with a U-loop. Then a T-type average

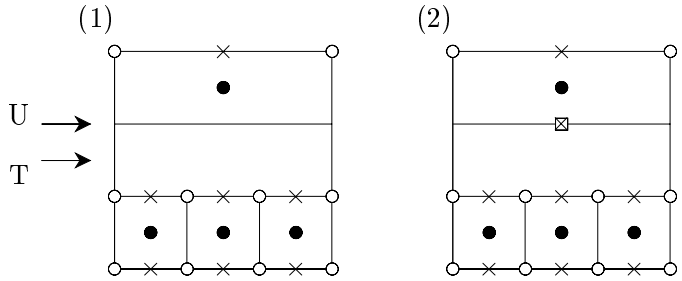

Figure 3.22: Procedure for calculating tracer advective velocities at a type-1 interface. Labeled arrows indicate interface rows. (1) Calculate AdvVel_Tn (cross-marks) as normal, skipping velocity interface row. (2) T-type average (boxed cross-mark) from the fine velocity row to the interface row.

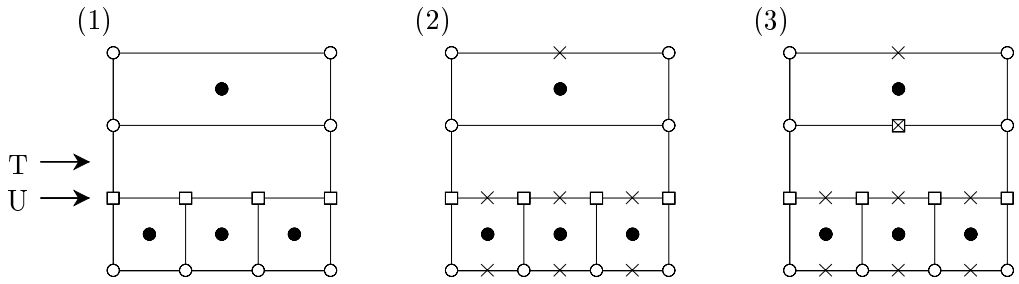

Figure 3.23: Procedure for calculating tracer advective velocities at a type-2 interface. Labeled arrows indicate interface rows. (1) U-type interpolation of meridional velocities from the coarse velocity row to the interface row. (2) Calculate AdvVel_Tn (cross-marks) as normal, skipping coarse velocity row. (3) T-type average (boxed cross-mark) from the velocity interface row to the coarse row.

is calculated from the fine velocity row to the interface row. This is shown schematically in Figure 3.22. With a type-2 interface, first an U-type interpolation is performed on the meridional velocities from the coarse velocity row to the interface row. Then AdvVel_Tn is computed with an S-loop. Finally a T-type average is calculated from the velocity interface row to the coarse row. This is shown schematically in Figure 3.23.

AdvVel_Tb is calculated from AdvVel_Te and AdvVel_Tn. The operations given above have provided these in the required locations. Thus, the vertical advective velocity is calculated in a T-loop. Its value is not necessary yet at the interface row and is left undefined there.

Advective Fluxes

Once the advective velocities are calculated, the advective fluxes given by (2.98) through (2.100) can be calculated. The advected quantities must be found by averaging neighboring tracer grid quantities. For AdvFlux_Te, the average $\overline{t_{i,j,k,\tau}}^{\lambda}$ is required, and for AdvFlux_Tb, the average $\overline{t_{i,j,k,\tau}}^{z}$ is required. Both are calculated as on the standard grid with a T-loop. The values are not required at the tracer interface row, so they are left undefined there. Then, both AdvFlux_Te and AdvFlux_Tb are calculated, also with a T-loop. These are also not required at the tracer interface row and are left undefined there.

For AdvFlux_Tn, the average $\overline{t_{i,j,k,\tau}}^{\phi}$ is required. Like AdvVel_Tn, this quantity is also a T_λ /U_ϕ quantity and is treated slightly different for type-1 and type-2 interfaces. For both interface types, the tracer quantity is first interpolated from the coarse tracer row to the interface row. Then the standard cell averages are calculated in a U-loop for a type-1 interface or an S-loop for a type-2 interface. Then AdvFlux_Tn is calculated using

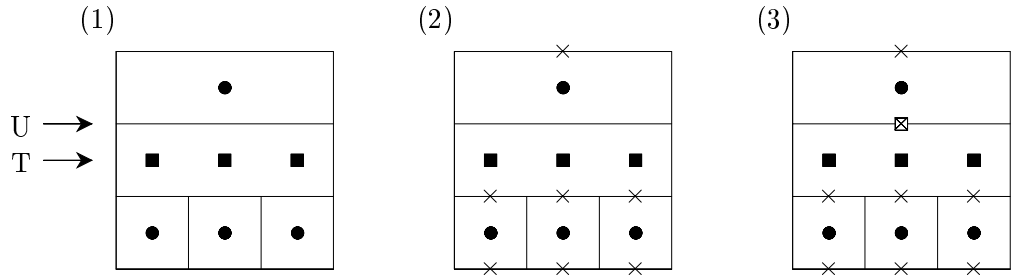

Figure 3.24: Procedure for calculating tracer advective fluxes at a type-1 interface. Labeled arrows indicate interface rows. (1) T-type interpolation of tracer grid quantity from the coarse tracer row to the interface row. (2) Calculate $\overline{t_{i,j,k,\tau}}^{\phi}$ and thus AdvFlux_Tn (cross-marks), skipping velocity interface row. (3) T-type average (boxed cross-mark) from the fine velocity row to the interface row.

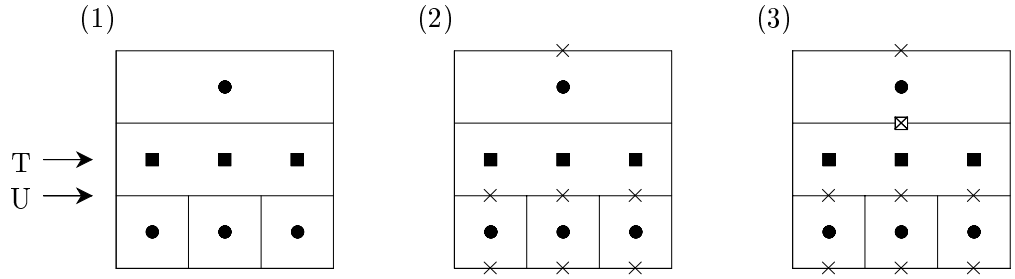

Figure 3.25: Procedure for calculating tracer advective fluxes at a type-2 interface. Labeled arrows indicate interface rows. (1) T-type interpolation of tracer grid quantity from the coarse tracer row to the interface row. (2) Calculate $\overline{t_{i,j,k,\tau}}^{\phi}$ and thus AdvFlux_Tn (cross-marks), skipping coarse velocity row. (3) T-type average (boxed cross-mark) from the fine velocity row to the interface row.

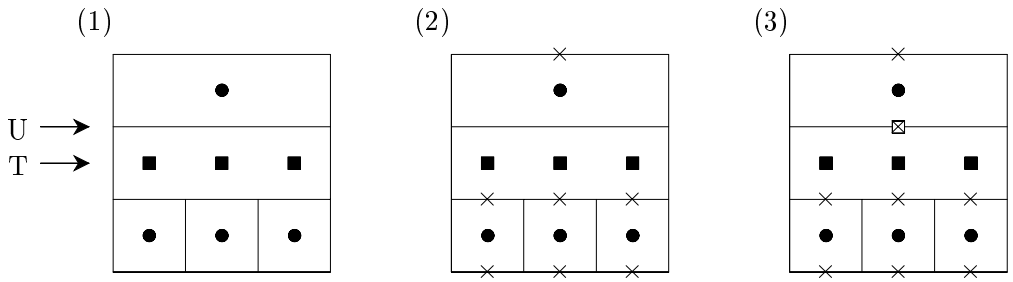

Figure 3.26: Procedure for calculating tracer diffusive fluxes at a type-1 interface. Labeled arrows indicate interface rows. (1) T-type interpolation of tracer grid quantity from the coarse tracer row to the interface row. (2) Calculate $\delta_\phi\left(t_{i,j,k,\tau}\right)$ and thus DiffFlux_Tn (cross-marks), skipping velocity interface row. (3) T-type average (boxed cross-mark) from the fine velocity row to the interface row.

these values along with the advective velocities, also in a U-loop and S-loop for type-1 and type-2 interfaces, respectively. Then a T-type average is performed on the fluxes from the fine velocity row to the interface row for a type-1 interface, and from the velocity interface row to the coarse row for a type-2 interface. These operations are shown schematically in Figure 3.24 and Figure 3.25.

Then, since the tracer advective fluxes are known at every necessary location, the time tendency of the tracer quantity due to advection, $ADV(t_{i,j,k,\tau})$, given by (2.101), is calculated with a T-loop. This quantity is not needed for the interface row and is left undefined there.

Diffusive Fluxes

For the diffusive term, the fluxes given by (2.102) through (2.104) must be calculated. For DiffFlux_Te, the average $\delta_\lambda\left(t_{i,j,k,\tau}\right)$ is required, and for DiffFlux_Tb, the average $\delta_z\left(t_{i,j,k,\tau}\right)$ is required. Both are calculated in a T-loop, with the unneeded interface values left undefined.

For DiffFlux_Tn, the difference $\overline{t_{i,j,k,\tau}}^{\phi}$ is required. Like AdvFlux_Tn, this quantity

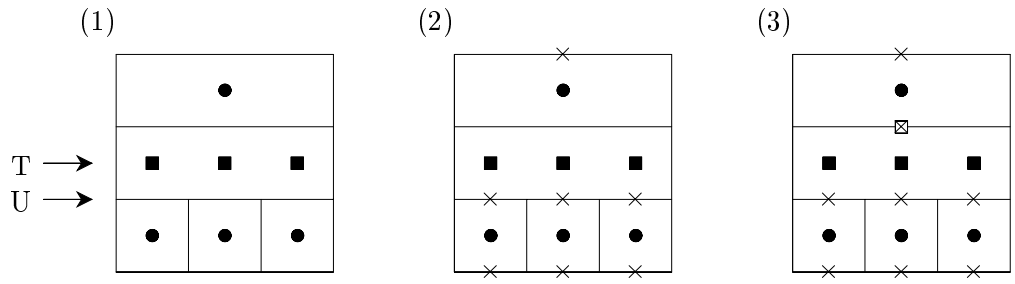

Figure 3.27: Procedure for calculating tracer diffusive fluxes at a type-2 interface. Labeled arrows indicate interface rows. (1) T-type interpolation of tracer grid quantity from the coarse tracer row to the interface row. (2) Calculate $\delta_\phi\left(t_{i,j,k,\tau}\right)$ and thus DiffFlux_Tn (cross-marks), skipping coarse velocity row. (3) T-type average (boxed cross-mark) from the fine velocity row to the interface row.

is also a T_λ /U_ϕ quantity and is treated slightly different for type-1 and type-2 interfaces. For both interface types, the tracer quantity is first interpolated from the coarse tracer row to the interface row. Note that this interpolation actually needs to be done only once for all of the tracer equation terms. Then the fluxes are calculated in a U-loop for a type-1 interface or an S-loop for a type-2 interface. Finally, a T-type average is performed on the fluxes from the fine velocity row to the interface row for a type-1 interface, and from the velocity interface row to the coarse row for a type-2 interface. These operations are shown schematically in Figure 3.26 and Figure 3.27.

Now that tracer diffusive fluxes are known at every necessary location, the time tendency of the tracer quantity due to diffusion $DIFF(t_{i,j,k,\tau})$, given by (2.105), is calculated with a T-loop. This quantity is not needed for the interface row and is left undefined there.

3.4.3 Baroclinic Momentum

The discretized momentum equations, given by (2.107) and (2.108), consist of advective, viscous, coriolis, metric, and pressure gradient terms. The general procedure for

advancing this equation on the standard grid compared to the procedure on a reduced grid as follows (column-only operations omitted):

Standard Grid	Reduced grid
	• Interpolate momentum quantities at interfaces.
• Calculate horizontal advective velocities.	• Calculate horizontal advective velocities.
	• Average meridional advective velocity at interfaces.
	• Interpolate or point-to-point copy tracer grid vertical velocity to interfaces.
• Calculate vertical advective velocity.	• Calculate vertical advective velocity.
• Calculate advective and diffusive fluxes.	• Calculate advective and diffusive fluxes.
	• Average meridional advective and diffusive fluxes at interfaces.
• Calculate divergences of advective and diffusive fluxes.	• Calculate divergences of advective and diffusive fluxes.
• Calculate coriolis and metric terms.	• Calculate coriolis and metric terms.
	• Interpolate or point-to-point copy density to interfaces.
• Calculate pressure gradient "fluxes."	• Calculate pressure gradient "fluxes."
	• Average meridional pressure gradient "fluxes" at interfaces.
• Calculate differences of pressure gradient "fluxes."	• Calculate differences of pressure gradient "fluxes."

Again it can be seen that the required modifications to the calculation procedure consist mainly of extra steps inserted between standard calculations and modifications to loops

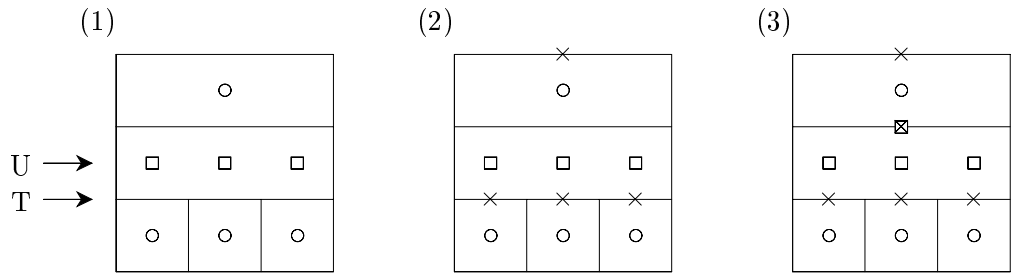

Figure 3.28: Procedure for calculating velocity grid advective velocities at a type-1 interface. Labeled arrows indicate interface rows. (1) U-type interpolation of meridional velocities from the coarse velocity row to the interface row. (2) Calculate AdvVel_Un (cross-marks) as normal, skipping coarse tracer row. (3) U-type average (boxed cross-mark) from the tracer interface row to the coarse row.

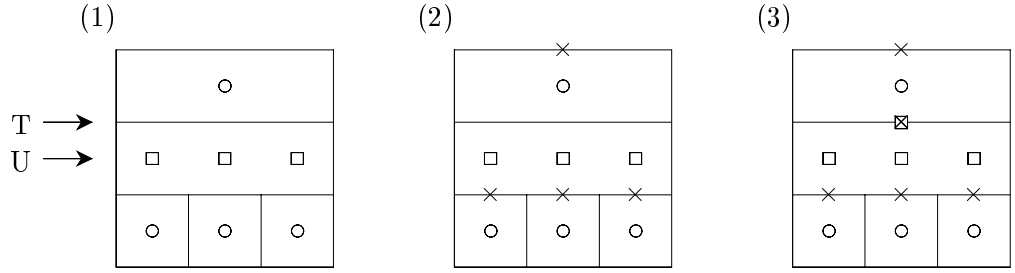

Figure 3.29: Procedure for calculating velocity grid advective velocities at a type-2 interface. Labeled arrows indicate interface rows. (1) U-type interpolation of meridional velocities from the coarse velocity row to the interface row. (2) Calculate AdvVel_Un (cross-marks) as normal, skipping tracer interface row. (3) U-type average (boxed cross-mark) from the fine tracer row to the interface row.

as described in Section 3.2.2. With this overview in mind, the details are given below.

Advective Velocities

Following Section 2.3.6, the advective velocities of (2.109) through (2.111) are first calculated. The advective velocity on the eastern face of the cell, AdvVel_Ue, is a U_ϕ quantity (see Section 3.3) and is calculated with a U-loop, which skips the velocity interface row. This quantity is not needed for the interface row, so it is left undefined there.

AdvVel_Un is a T_ϕ /U_λ quantity and is treated slightly different for type-1 and type-

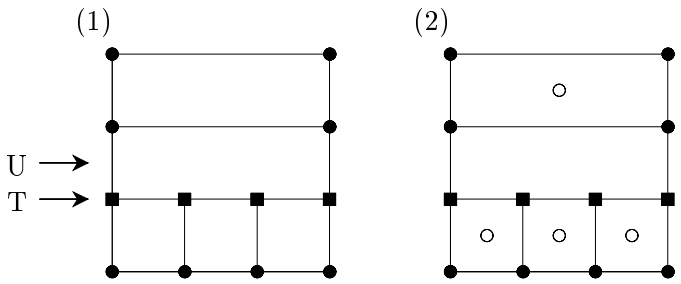

Figure 3.30: Procedure for calculating velocity grid bottom advective velocities at a type-1 interface. Labeled arrows indicate interface rows. (1) T-type interpolation of tracer grid vertical velocities from the coarse tracer row to the interface row. (2) Calculate averages of neighoring tracer vertical velocities, skipping velocity interface row.

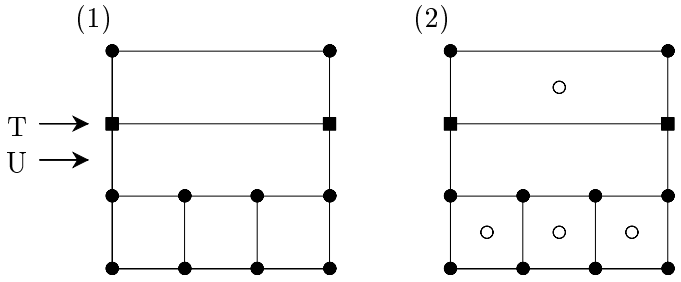

Figure 3.31: Procedure for calculating velocity grid bottom advective velocities at a type-2 interface. Labeled arrows indicate interface rows. (1) T-type point-to-point copy of tracer grid vertical velocities from the fine tracer row to the interface row. (2) Calculate averages of neighoring tracer vertical velocities, skipping velocity interface row.

2 interfaces. For both types, the meridional velocity is first interpolated with from the coarse velocity row to the interface row. With a type-1 interface, velocities are calculated in an S-loop, while a T-loop is used with a type-2 interface. Then a U-type average is performed from the tracer interface row to the coarse row for a type-1 interface and from the fine tracer row to the interface row for a type-2 interface. These operations are shown schematically in Figure 3.28 and Figure 3.29.

AdvVel_Ub is calculated from AdvVel_Ue, AdvVel_Un, and AdvVel_Tb, by (2.111). The operations already given have provided the horizontal advective velocities in the required locations. For the vertical advective velocity at the bottom, an average over

the surrounding tracer vertical advective velocities is computed. With a type-1 interface, the tracer vertical advective velocities are first interpolated from the coarse tracer row to the interface. With a type-2 interface, they are point-to-point copied from the fine tracer row to the interface row. Then averages are calculated on each velocity point of the neighboring tracer vertical velocities using a U-loop. The value at the velocity interface is not necessary and is left undefined. These operations are shown schematically in Figure 3.30 and Figure 3.31.

Advective Fluxes

Having obtained the velocity grid advective velocities, the advective fluxes given by (2.112) through (2.117) can be calculated. The zonal and vertical fluxes, AdvFlux_Ue, AdvFlux_Ve, AdvFlux_Ub, and AdvFlux_Vb, are computed with a U-loop, which skips the velocity interface row. The are not needed at the interface row, so they are left undefined there.

For AdvFlux_Un and AdvFlux_Vn, the averages $\overline{u_{i,j,k,\tau}}^{\phi}$ and $\overline{v_{i,j,k,\tau}}^{\phi}$ are required. Like AdvVel_Un and AdvVel_Vn, they are T_ϕ /U_λ quantities and are treated slightly different for type-1 and type-2 interfaces. The procedure follows AdvVel_Un directly. See the description of its calculation and Figure 3.28 and Figure 3.29 for details.

Then, since the advective momentum fluxes are known at the necessary locations, the time tendency of the momentum quantities due to advection, $ADV(u_{i,j,k,\tau})$ and $ADV(v_{i,j,k,\tau})$, given by (2.118) and (2.119), are calculated in a U-loop. These quantities are not needed for the interface row and are left undefined there.

Diffusive Fluxes and Diffusive Metric Terms

For the viscous terms, the fluxes given by (2.120) through (2.125) must be calculated. For DiffFlux_Ue and DiffFlux_Ve, the differences $\delta_\lambda\left(u_{i,j,k,\tau}\right)$ and $\delta_\lambda\left(v_{i,j,k,\tau}\right)$ are required. For DiffFlux_Ub and DiffFlux_Vb, the differences $\delta_z\left(t_{i,j,k,\tau}\right)$ and $\delta_z\left(t_{i,j,k,\tau}\right)$ are required. All four are calculated in a U-loop, with the unneeded interface values left undefined.

For DiffFlux_Un and DiffFlux_Vn, the differences $\delta_\phi\left(u_{i,j,k,\tau}\right)$ and $\delta_\phi\left(v_{i,j,k,\tau}\right)$ are required. Like AdvVel_Un, AdvVel_Vn, AdvFlux_Un, and AdvFlux_Vn, they are $T_\phi\ /U_\lambda$ quantities and are treated slightly different for type-1 and type-2 interfaces. The procedure follows that of AdvVel_Un directly. See the description of its calculation and Figure 3.28 and Figure 3.29 for details.

The viscous metric terms, given by (2.128) and (2.129), are calculated with a U-loop, skipping the interface rows where their values are unneeded.

Then, since the diffusive momentum fluxes and their metric terms are known at every necessary location, the time tendency of the momentum quantities due to viscosity, $VISC(u_{i,j,k,\tau})$ and $VISC(v_{i,j,k,\tau})$, given by (2.126) and (2.127), are calculated with a U-loop. These quantities are not needed for the interface row and are left undefined there.

Coriolis and Metric Terms

The coriolis terms, given by (2.130) and (2.131), being particular simple, are calculated in a U-loop, skipping the interface rows where their values are unneeded. The metric terms, given by (2.132) and (2.133), are handled in this manner as well.

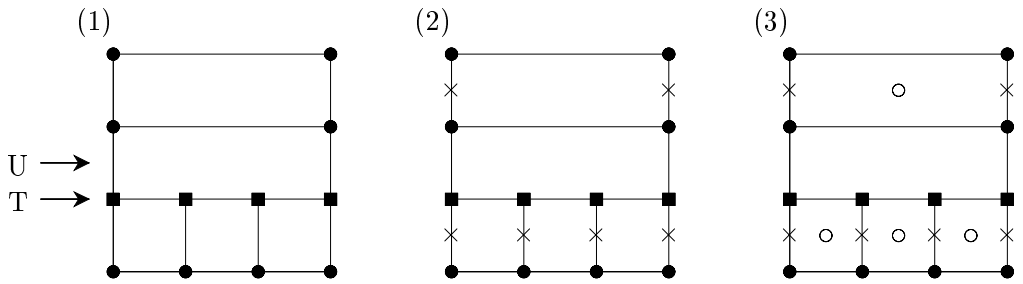

Figure 3.32: Procedure for calculating longitudinal pressure gradient at a type-1 interface. Labeled arrows indicate interface rows. (1) T-type interpolation of densities from the coarse tracer row to the interface row. (2) Calculate latitudinal (and depth) averages, skipping velocity interface row. (3) Calculate pressure gradient, skipping velocity interface row.

Pressure Gradients

The pressure gradient terms, given by (2.134) and (2.135) at the surface and (2.136) and (2.137) at deeper levels, are velocity grid quantities that can be cast into a conservative flux form. They consist of gradients in one direction of averages taken over the orthogonal direction(s). Thus, a conservative form will treat the averages as flux quantities which must be the same for each cell sharing the boundary on which the flux is located. This is analogous to the derivatives of fluxes which comprise the advective and diffusive terms already presented. Latitudinal gradients will require an average of the flux quantities for conservation.

The calculation of the pressure gradients with type-1 interfaces begins with a T-type interpolation of densities, which are tracer grid quantities, from the coarse tracer row to the interface row. Latitudinal averages are computed in a U-loop and include depth averaging for all but the top layer. Longitudinal averages are computed in an S-loop, again with depth averaging for all but the top layer, then averaged with a U-type average from the tracer interface row to the coarse row. Finally, both the latitudinal and longitudinal pressure gradients are calculated in a U-loop, with the unneeded interface row

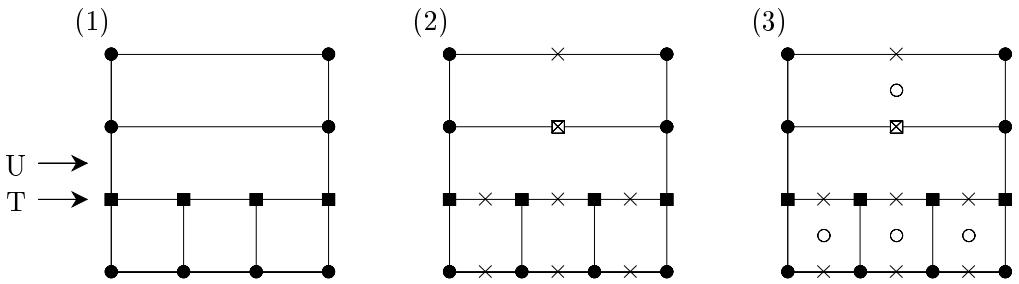

Figure 3.33: Procedure for calculating latitudinal pressure gradient at a type-1 interface. Labeled arrows indicate interface rows. (1) T-type interpolation of densities from the coarse tracer row to the interface row. (2) Calculate longitudinal (and depth) averages, skipping coarse tracer row. U-type average of resulting fluxes from tracer interface row to coarse row. (3) Calculate pressure gradient, skipping velocity interface row.

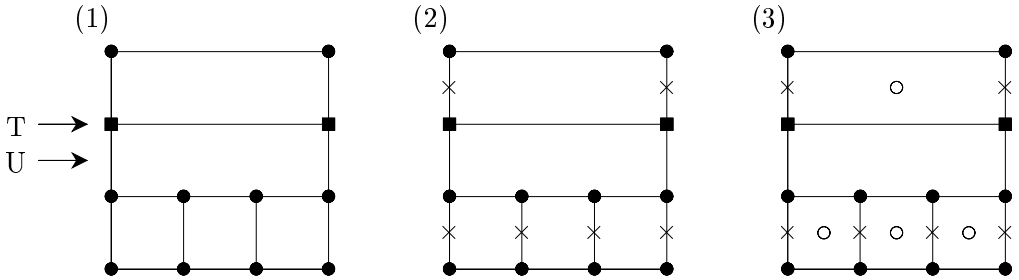

Figure 3.34: Procedure for calculating longitudinal pressure gradient at a type-2 interface. Labeled arrows indicate interface rows. (1) T-type point-to-point copy of densities from the fine tracer row to the interface row. (2) Calculate averages, skipping velocity interface row. (3) Calculate pressure gradient, skipping velocity interface row.

left undefined. These operations are shown schematically in Figure 3.32 and Figure 3.33.

The calculation of the pressure gradients with type-2 interfaces begins with a T-type point-to-point copy of densities from the fine tracer row to the interface row. Latitudinal averages are computed in a U-loop and include depth averaging for all but the top layer. Longitudinal averages are computed in an S-loop, again with depth averaging for all but the top layer, then averaged with a U-type average from the fine tracer row to the interface row. Finally, both the latitudinal and longitudinal pressure gradients are calculated in a U-loop, with the unneeded interface row left undefined. These operations are shown schematically in Figure 3.34 and Figure 3.35.

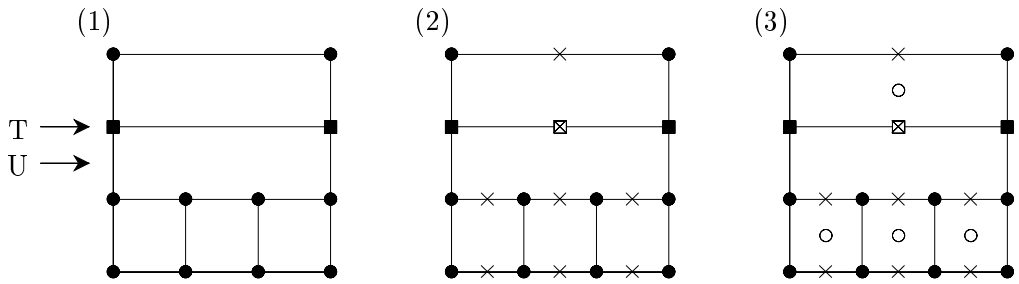

Figure 3.35: Procedure for calculating latitudinal pressure gradient at a type-2 interface. Labeled arrows indicate interface rows. (1) T-type point-to-point copy of densities from the fine tracer row to the interface row. (2) Calculate averages, skipping tracer interface row. U-type average of resulting fluxes from fine tracer row to interface row. (3) Calculate pressure gradient, skipping velocity interface row.

3.4.4 Barotropic Momentum and Surface Height

The discretized barotropic momentum and surface height equations, given by (2.138) through (2.140), consist of surface height gradient, velocity divergence, coriolis, and forcing terms. The general procedure for advancing this equation on the standard grid compared to the procedure on a reduced grid as follows:

Standard Grid	Reduced grid
	• Interpolate meridional and zonal barotropic velocities, or point-to-point copy zonal barotropic velocity at interfaces.
• Calculate "fluxes," i.e. averages, of barotropic velocities	• Calculate "fluxes," i.e. averages, of barotropic velocities
	• Average meridional velocity "fluxes" at interfaces.
• Calculate differences of velocity "fluxes" for surface height equation.	• Calculate differences of velocity "fluxes" for surface height equation.
	• Interpolate or point-to-point copy surface height at interfaces.
• Calculate "fluxes," i.e. averages, of surface height.	• Calculate "fluxes," i.e. averages, of surface height.
	• Average surface height "fluxes" at interfaces.
• Calculate differences of surface height "fluxes" for barotropic momentum equations.	• Calculate differences of surface height "fluxes" for barotropic momentum equations.
• Calculate surface height filter, if necessary.	• Calculate surface height filter, if necessary.

By this comparison it can be seen that the required modifications to the calculation procedure consist mainly of extra steps inserted between standard calculations and modifications to loops as described in Section 3.2.2. With this overview in mind, the details are given below.

Surface Height Gradients

The surface height gradient terms, given by (2.145) and (2.146), are velocity grid quantities that, like the pressure gradient terms of the baroclinic momentum equations,

can be cast into a conservative flux form.

The calculation of gradients with type-1 interfaces begins with a T-type interpolation of surface height, which is a tracer grid quantity, from the coarse tracer row to the interface row. Latitudinal averages are computed in a U-loop, and longitudinal averages are computed in an S-loop and a U-type average from the tracer interface row to the coarse row computed. Finally, both the latitudinal and longitudinal gradients are calculated in a U-loop, with the unneeded interface row left undefined. See the schematics of the similar operation on the pressure gradient terms, Figure 3.32 and Figure 3.33.

The calculation of gradients with type-2 interfaces begins with a T-type point-to-point copy of surface height from the fine tracer row to the interface row. Latitudinal averages are computed in a U-loop, and longitudinal averages are computed in an S-loop and a U-type average from the fine tracer row to the interface row computed. Finally, both the latitudinal and longitudinal gradients are calculated in a U-loop, with the unneeded interface row left undefined. See the schematics of the similar operation on the pressure gradient terms, Figure 3.34 and Figure 3.35.

These two gradients are all that is needed for the update of the surface height, which is performed in a T-loop.

Barotropic Velocity Gradients

The barotropic velocity gradients are similar to the surface height gradients, but are calculated at tracer grid points. Again, averages are calculated, then differences are taken.

For a type-1 interface, the first step is a U-type point-to-point copy of the zonal barotropic velocity from the fine velocity row to the interface row. Latitudinal averages of zonal velocity are calculated in a T-loop, while longitudinal averages of meridional velocity

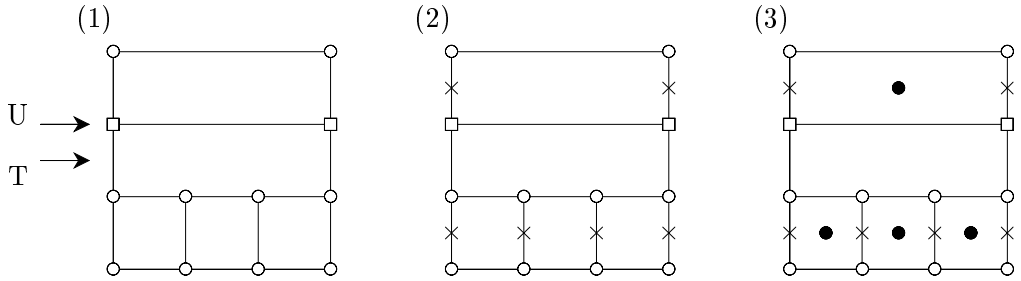

Figure 3.36: Procedure for calculating zonal barotropic velocity gradient at a type-1 interface. Labeled arrows indicate interface rows. (1) U-type point-to-point copy of zonal barotropic velocity from the fine velocity row to the interface row. (2) Calculate latitudinal averages, skipping tracer interface row. (3) Calculate zonal gradient, skipping tracer interface row.

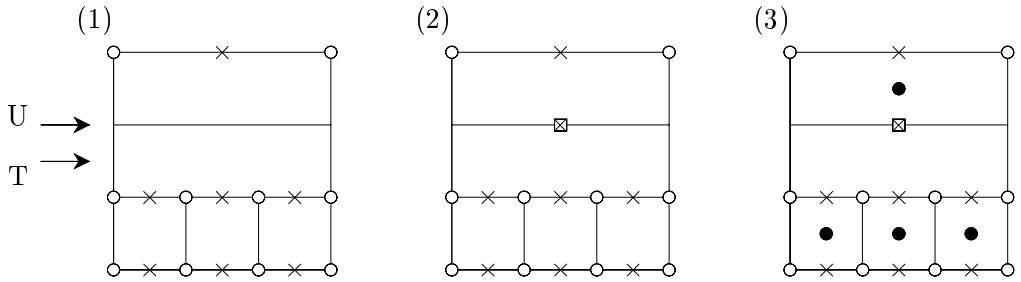

Figure 3.37: Procedure for calculating meridional barotropic velocity gradient at a type-1 interface. Labeled arrows indicate interface rows. (1) Calculate longitudinal averages, skipping velocity interface row. (2) T-type average of resulting fluxes from fine velocity row to interface row. (3) Calculate meridional gradient, skipping tracer interface row.

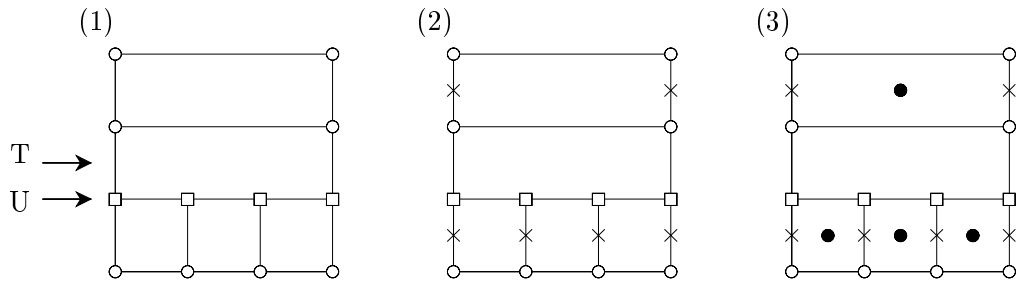

Figure 3.38: Procedure for calculating zonal barotropic velocity gradient at a type-2 interface. Labeled arrows indicate interface rows. (1) U-type interpolation of zonal barotropic velocity from the coarse velocity row to the interface row. (2) Calculate latitudinal averages, skipping tracer interface row. (3) Calculate zonal gradient, skipping tracer interface row.

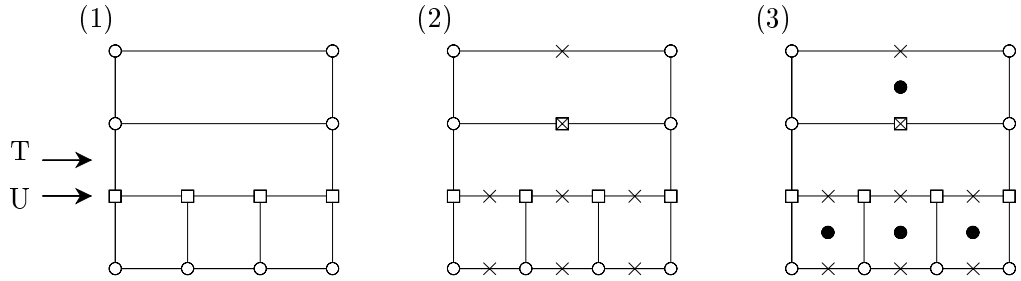

Figure 3.39: Procedure for calculating meridional barotropic velocity gradient at a type-2 interface. Labeled arrows indicate interface rows. (1) U-type interpolation of meridional barotropic velocity from the coarse velocity row to the interface row. (2) Calculate longitudinal averages, skipping coarse velocity row. T-type average of resulting fluxes from velocity interface row to coarse row. (3) Calculate meridional gradient, skipping tracer interface row.

are calculated in a U-loop. Then the longitudinally averaged meridional velocity is T-type averaged from the fine velocity row to the interface row. Finally, both of the gradients are calculated in a T-loop, with the unneeded interface row left undefined. These operations are shown schematically in Figure 3.36 and Figure 3.37.

For a type-2 interface, the first step is a U-type interpolation of both the zonal and meridional barotropic velocities from the coarse velocity row to the interface row. Latitudinal averages of zonal velocity are calculated in a T-loop, while longitudinal averages of meridional velocity are calculated in an S-loop. Then the longitudinally averaged merid-

ional velocity is T-type averaged from the velocity interface row to the coarse row. Finally, both of the gradients are calculated in a T-loop, with the unneeded interface row left undefined. These operations are shown schematically in Figure 3.38 and Figure 3.39.

Coriolis and Forcing Terms

The coriolis terms given by (2.143) and (2.144), being particularly simple in form, are calculated in a U-loop, skipping the interface rows where their values are unneeded. The forcing terms given by (2.141) and (2.142) are calculated in this manner as well.

Surface Height Filter

The surface height filter of Section 2.3.8 is modified in a different manner for use with the reduced grid. The computational mode that it is designed to suppress is defined by the grid resolution. On the reduced grid, the resolution differs between latitude rows at interfaces. So the filter must be modified not to operate across interfaces, and therefore between grids of different resolution. This is accomplished by treating the interface dummy row as a land row for the purposes of the filtering calculation. The filter conserves the surface height at land, so this method still conserves locally and globally. This effectively separates the regions of differing resolution in a simple manner, allowing the calculations described in Section 2.3.8 to be used with only the normal loop modifications of Section 3.2.2.

3.5 Parallel Implementation

In order to avoid major changes to the model, the decomposition of the reduced grid must work within the existing framework. Longitudinal decomposition is done as

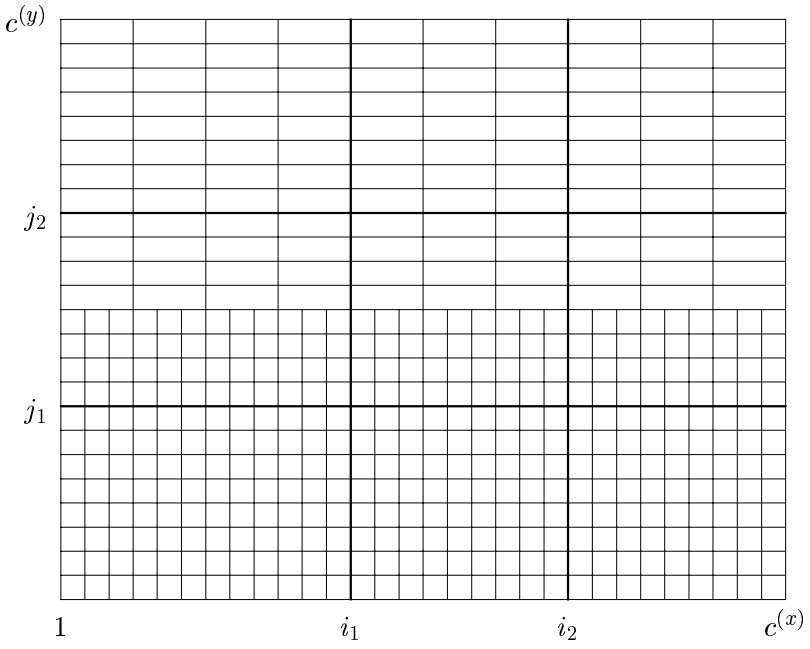

Figure 3.40: An example of reduced grid domain decomposition for nine subdomains.

before with the restrictions that arise due to the fewer cells in the zonal direction in some regions. Since the grid is coarsened at some latitudes, the domain must be broken up into subdomains which still have whole numbers of the coarsest cells. Using Figure 3.40 as an example, the domain cannot be broken longitudinally into three equally sized subdomains, even though a similar standard grid could be made into three subdomains with 10 cells each in longitude. While arbitrary resolutions could present a problem with load balance because of this, in practice the resolution can be selected to give flexibility in decomposing the domain in this dimension.

Latitudinal decomposition is accomplished with a modified algorithm. Again, no ghost cell borders are included in these calculations.

Algorithm 3.5.1. Reduced grid domain decomposition
Given:
$c^{(y)}$ {the number of cells in the latitudinal direction}
$c_j^{(x)}$ {the number of cells in the longitudinal direction at latitude index j, for $j = 1, \ldots, c^{(y)}$}
s {the number of subdomains over which to decompose the domain latitudinally}
w_j {a weight whose value is proportional to the relative amount of work necessary for the number of cells, $c_j^{(x)}$, at latitude row j, for $j = 1, \ldots, c^{(y)}$. The absolute value of the weight is not significant—only the relative value}
Calculate:
$j \Leftarrow 1$
$W_{\text{avg}} \Leftarrow \frac{1}{s} \sum_{j'=1}^{c^{(y)}} w_{j'}$
for $n = 1$ to $s - 1$
$t \Leftarrow 0$
$d_n \Leftarrow 0$
repeat
$t \Leftarrow t + w_j$
$d_n \Leftarrow d_n + 1$
$j \Leftarrow j + 1$
until $t \geq W_{\text{avg}}$
$d_s \Leftarrow c^{(y)} - j + 1$
Then each subdomain, n, will contain the points given by the indices

$$i = i' + \sum_{n'=1}^{n-1} d_{n'} \text{ for } i' = 1, \ldots, d_n,$$

where the valid range of indices is 1 to $c^{(y)}$, and any ghost cells which may be needed for the numerical method employed have not been included.

The weights, w_j, must be set empirically, but some reasonable bounds can be placed on them. At one extreme, if the reduction in the number of cells from coarsening the grid did not lessen the work, all of the weights would be set to a constant. At the other extreme, with the reduction in work equaling the theoretical maximum, w_j would be proportional to the number of cells at each latitude. In practice, the correct choice was subject to trial-and-error, and a slightly different code implementation would undoubtedly have a better optimum method. One of the choices which gave good results for some decompositions was to set

$$w_j = n_j^{\frac{\log 2}{\log 3}}, \tag{3.4}$$

where n_j is the number of reduced grid cells at latitude row j.

Additionally there is a restriction due to the implementation of the interfaces by "dummy" rows (see Section 3.2.1.) A subdomain boundary may not be too near a grid interface without introducing complexities which would require reorganization of the code and algorithms. Since this is explicitly against the design goals, interfaces are restricted such that the "dummy" rows may not be located at a subdomain boundary. This places a limitation on the available decompositions for fixed numbers of subdomains. When the number of desired subdomains increases, the available decompositions which meet the above criteria may be few. However, this limitation is rarely an issue for the numbers of processors which can efficiently be used at a given resolution.

Chapter 4

Limited Basin Simulations

Since the targeted application of the reduced grid method is the global ocean domain, the final test of the method will be in this context. However, the time necessary for integration of the model equations and the complexities of flow and topography make the full global runs an impractical forum for looking at the detailed influence of the method. The tests described in this chapter start at the simplest level possible that will exercise the complete model. The comparatively short execution time of the first two tests enable the examination of a large number of options. Proceeding to the last test of this chapter, the execution times increase dramatically, and the number of options tested is greatly reduced.

Level	Depth to Tracer and U,V point	Depth to bottom of cell	Thickness
1	50	100	100
2	223.57	347.14	247.14
3	607.08	867.01	519.87
4	1313.8	1760.5	893.48
5	2380.2	3000	1239.5

Table 4.1: Basin depth levels. All units are meters

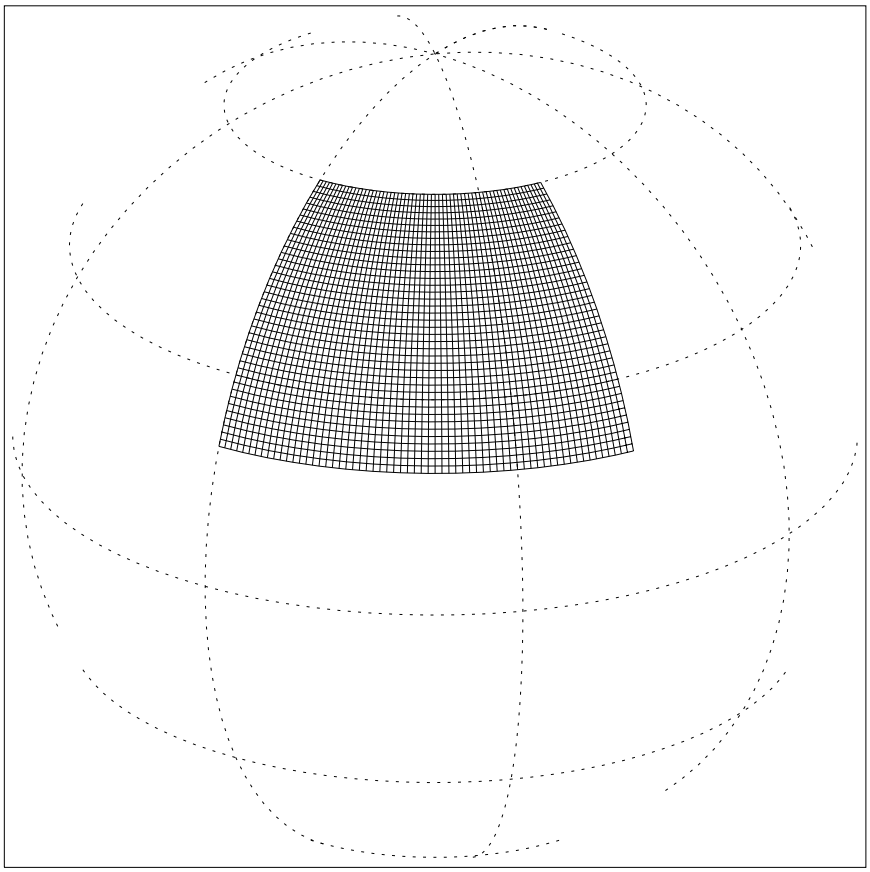

Figure 4.1: The horizontal grid for the standard resolution case of the midlatitude basin.

Parameter	value	Parameter	value
Horiz. Viscosity	10^9 cm^2/s	Basin width	63 degrees
Vert. Viscosity	20 cm^2/s	Basin height	40 degrees
Baroclinic step	60 min.	Basin depth	3000 m
Barotropic step	2.5 min.		

Table 4.2: Parameter values for the linear, wind-forced basin.

The first choice in simplifying the model runs is the choice of domain. By using a flat bottomed, rectangular domain with vertical walls at the boundaries, the effects of varying topography are eliminated. The tests of this chapter will all use the same topography and only vary the forcing applied. The basin, shown in Figure 4.1, is in the northern hemisphere with vertical walls at 20 degrees and 60 degrees latitude and a width of 63 degrees with vertical walls on the east and west as well. It has five vertical layers, with the vertical grid defined as in Table 4.1.

4.1 Wind-Driven, Flat Bottomed Basin

The long time scales in the model equations are due to the thermohaline forcing, the forcing due to the density variations caused by varying temperature and salinity. By fixing the temperature and salinity of the model domain at spatially uniform values, the tracer transport equations and the pressure gradient terms due to density variations can be effectively removed from the runs without altering the code. This allows much shorter runs to be accomplished at first. Two sets of simulations will be done with this "constant tracers" condition.

In a simulation with spatially uniform tracer values, the only way to create a circulation is via the momentum flux at the ocean surface. Again, for simplicity, this is

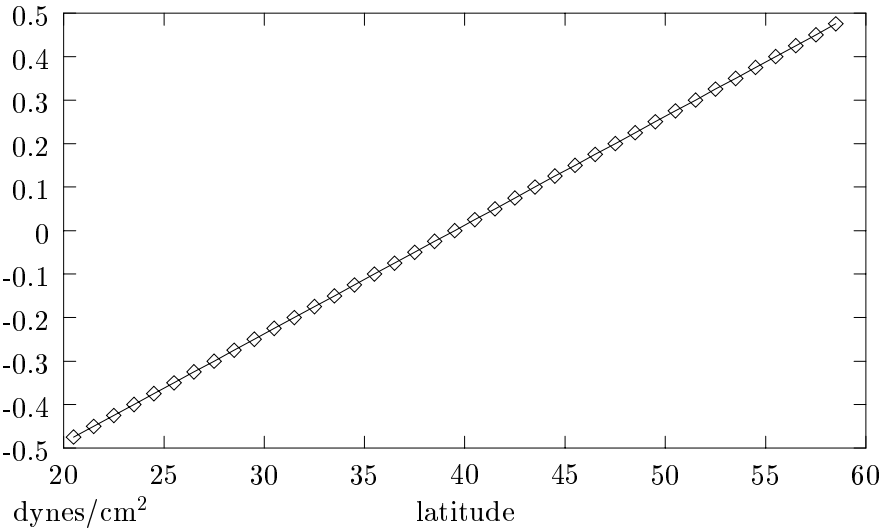

Figure 4.2: Surface wind stress for the linear-wind-forced basin. The forcing is zonally constant, with a zero meridional component.

Case	Longitudinal spacing	Latitudinal spacing
Fine	0.5	0.5
Standard	1.0	1.0
Coarse	3.0	1.0
Reduced	1.0 / 3.0	1.0

Table 4.3: Resolution of the four basin cases. Values are degrees.

assumed to be in the zonal direction only and to have variation in the latitudinal direction only. Figure 4.2 shows the wind stress used. Because this forcing is linear, I will refer to this test case as the linear-wind-forced basin. Various model parameters are shown in Table 4.2. The resulting velocity field is expected to be a clockwise gyre with a narrowed and intensified western boundary current, as described by Pedlosky in [51] and [52]. This case and the following is comparable in character to that used in [10] and [54].

Three different cases of the standard grid model will be shown. One case, called the "fine grid", has a resolution of 0.5 degree in both longitude and latitude, giving it a total of 126 by 80 horizontal cells. This case will be used as a reference result to compare the

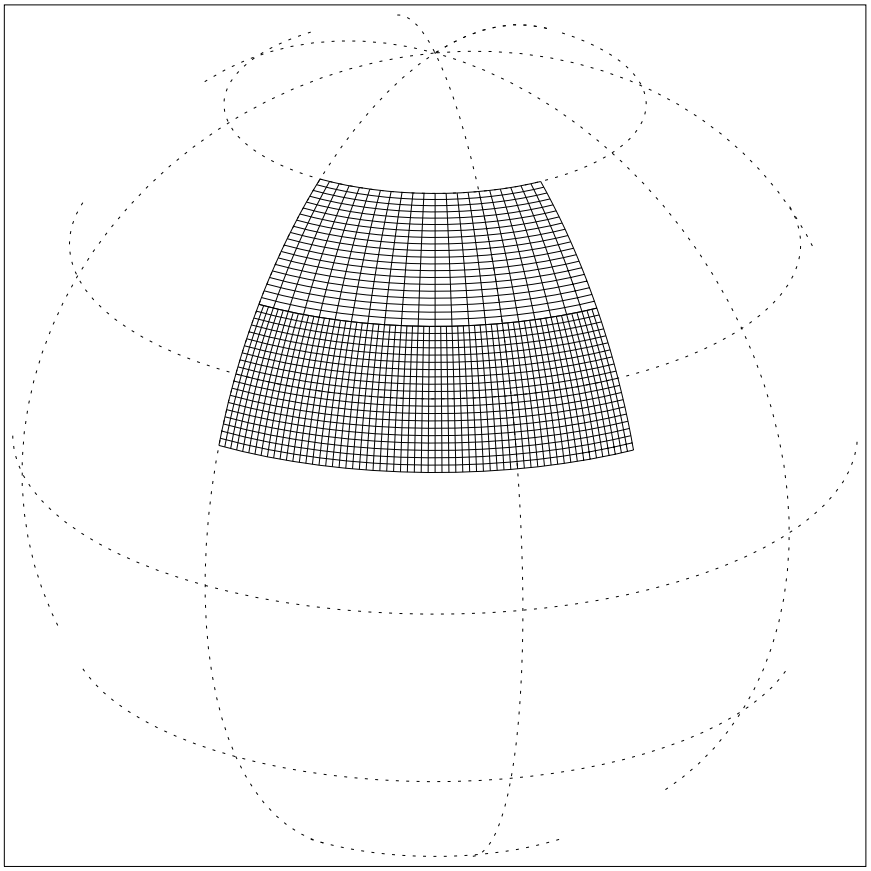

Figure 4.3: The horizontal grid for the reduced grid case of the midlatitude basin.

others against. The second case, called the "standard grid", has a resolution of 1 degree in both longitude and latitude, giving it a total of 63 by 40 horizontal cells. The third case, called the "coarse grid", has a latitudinal resolution of 1 degree as well, but a longitudinal resolution of 3 degrees, giving a total of 21 by 40 horizontal cells. Table 4.3 summarizes the resolutions of these cases.

Ten versions of the "reduced grid" case are presented, all having a resolution of 1 degree in latitude but a longitudinal resolution which is 1 degree in the southern half of the domain and 3 degrees in the northern half of the domain. Thus this grid has an interface separating a region with resolution similar to the coarse grid case from a region with res-

olution like that of the standard grid case. This is illustrated in Figure 4.3. The tracer interface is located at 40 degrees latitude. The various reduced grid simulations differ in both the type of interface used (see Section 3.1.3) and the method used to interpolate the coarse grid variables to the fine grid.

The time steps used for the standard, coarse, and reduced grid resolution cases are given in Table 4.2. The time step is limited by the latitudinal spacing for this simulation. Thus the standard, coarse, and reduced grid cases, all having a latitudinal spacing of 1 degree, all use the same time steps. The fine case, which has a latitudinal grid spacing half as large as the others, uses time steps of half the size of those given in Table 4.2.

Previous work in nested primitive equation models has included linear, quadratic polynomial, and cubic spline interpolation of model variables and/or fluxes at grid interfaces (see [26], [30], and [76]). Here, four different types of polynomial interpolation are used to obtain fine grid values from the coarse grid values at the interface. They represent no interpolation, linear interpolation, quadratic interpolation, and cubic interpolation (see Section A.4.1 for details). In addition, cubic spline interpolation is used in these cases. No interpolation corresponds to using the average of neighboring quantities as the interface value. As Section A.1 shows, this leads to conservation of second moments, which is a potentially important property. However, this method is not expected to give good results, as Section A.2 shows that it leads to poor accuracy at the interfaces.

4.1.1 Kinetic Energy

For each case, the model is run for 150 simulated days, which is enough to bring the solution nearly to a steady state. A comparison of the final volume averaged kinetic energy over the whole basin for the various cases is shown in Table 4.4. The coarse case

Grid	Int. Type	Interpolation	K. E. dynes/cm^3	% "error"
Coarse	n/a	n/a	0.01664	18
Reduced	1	none	0.01384	-1.9
Reduced	1	linear	0.01458	3.3
Reduced	1	quadratic	0.01437	1.8
Reduced	1	cubic	0.01431	1.4
Reduced	1	cubic spline	0.01444	2.3
Reduced	2	none	0.01453	3.0
Reduced	2	linear	0.01429	1.3
Reduced	2	quadratic	0.01433	1.6
Reduced	2	cubic	0.01424	0.9
Reduced	2	cubic spline	0.01424	0.9
Standard	n/a	n/a	0.01423	0.9
Fine	n/a	n/a	0.01411	n/a

Table 4.4: Steady-state kinetic energy of the basin with linear wind forcing for various interpolation orders, interface types, and grid resolutions.

differs from the fine case by the most, with nearly 18 percent more energy. The standard resolution case is the closest, as expected, with less than 1 percent error. The reduced grid cases all have an error on the order of a few percent, with all but the type-1 case with no interpolation having too much kinetic energy. This tendency of the reduced grid to have too high a value is expected from the results of the coarse grid, since the resolution of these two grids is the same for half the region.

The kinetic energy is shown as a function of time for the various resolutions in Figure 4.4. The large difference in the coarse case is immediately evident. The nearly coincident curves for the remaining resolutions illustrates the small size of the differences between them and shows the similarity in their temporal behavior.

The kinetic energy as a function of time for the various reduced grid interface conditions is shown in Figure 4.5 for a type-1 interface and Figure 4.6 for a type-2 interface, with the fine resolution case shown for reference. The reduced grid results show very good agreement with the fine case, the exception being the type-1 case with no interpolation.

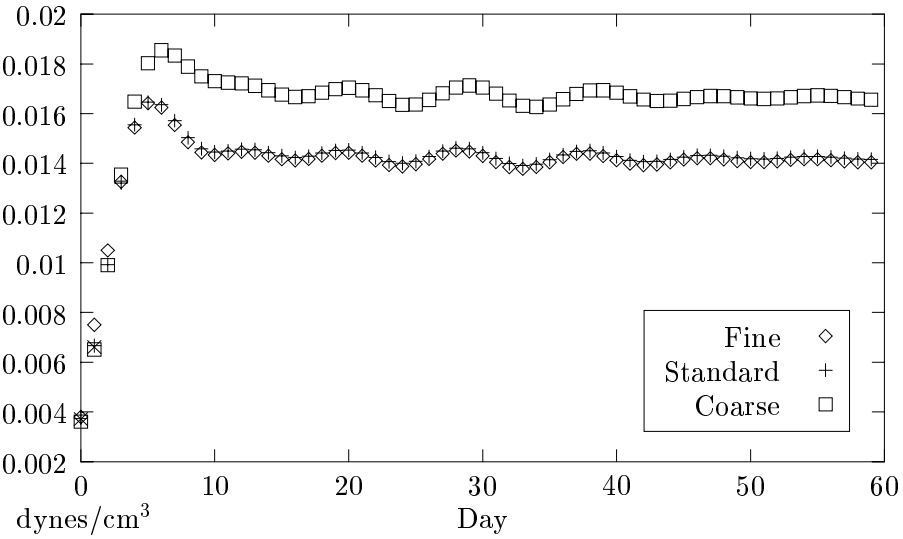

Figure 4.4: Volume averaged kinetic energy for the various grids with linear wind forcing.

The temporal behavior in that case is sluggish, i.e. the size of the fluctuations is smaller, when compared to the others.

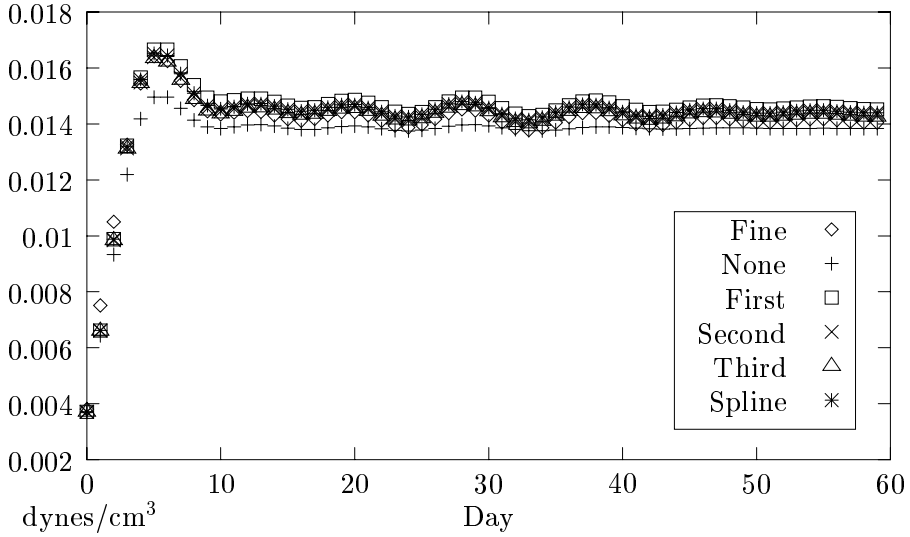

Figure 4.5: Volume averaged kinetic energy of the linear-wind-forced basin with a type-1 reduced grid interface by interpolation type. The fine resolution case is shown for comparison.

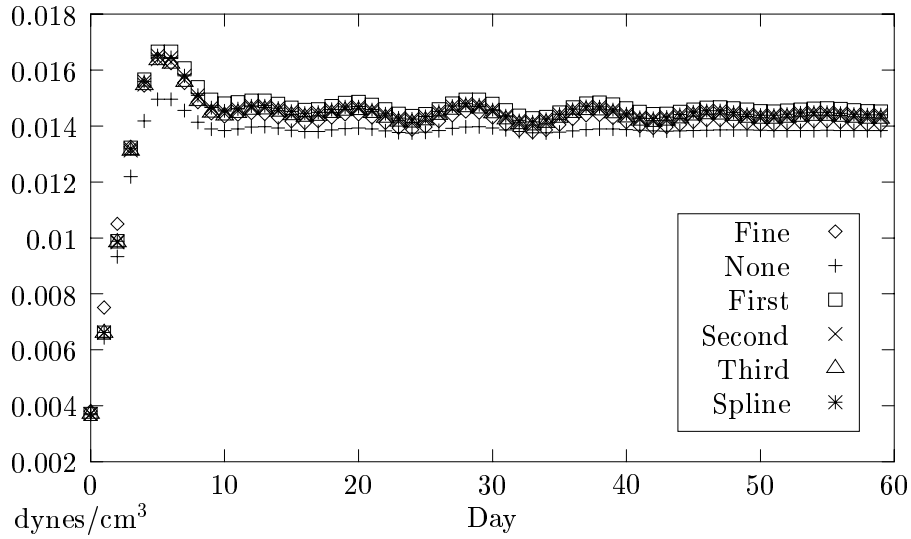

Figure 4.6: Same as Figure 4.5 except for a type-2 interface.

4.1.2 Surface Height

Figure 4.7 shows the surface height of the fine resolution case. There is a maximum in the surface height which is located near the western boundary, and the minimum is located in the northwestern corner of the basin. The zonal gradients across the western edge are the strongest. It is in this region, where resolution will likely be important, that the largest differences should be seen in the other cases.

Figure 4.8 shows the surface height of the standard resolution case. A comparison to the fine resolution case was made by averaging the fine resolution surface height to the resolution of the standard case. For this particular example, each 2 by 2 set of cells from the fine grid surface height was averaged to the corresponding single cell of the standard grid. Similar averaging will be done for all of the remaining cases as well. Then the resulting surface height was subtracted from that of the standard grid to obtain an "error" in the standard resolution surface height, which is shown in Figure 4.9. As expected, the largest differences are at the western boundary. The standard surface height is too low in the region of the maximum and too high at the westernmost edge, next to the wall. Likewise it is also too high at the northern and southern edges. Throughout most of the central and eastern regions the difference is smaller. A closer look will be made later, after the other cases are shown.

Figure 4.10 shows the surface height of the coarse resolution case. Again, a comparison to the fine resolution case was made by averaging the fine resolution surface height to the resolution of the coarse case, this time requiring 6 by 2 cell averages. The resulting difference is plotted in Figure 4.11. The pattern of the differences is very similar to that of the standard case, but the magnitude of the differences have increased by about an order

of magnitude.

The influence of resolution in the western boundary region is very pronounced in this test, so the effects of the coarsening of resolution across the mid-basin interface in the reduced grid cases should be easy to observe. An example case is shown in Figure 4.12 and Figure 4.13, showing the surface height and the difference of the surface height compared to the fine case for the type-1 interface with linear interpolation. This time the averaging of the fine resolution case for comparison was accomplished as with the standard grid for the southern half of the basin and the coarse grid for the northern half.

One feature common to all of the reduced grid cases is the relatively large difference observed along the western boundary in the northern half of the basin. This is comparable to that found in the coarse case, but it extends no more than one cell into the southern half of the basin. In the southern half, the difference observed along the western boundary is comparable to that in the standard case. Another common feature is the relatively smaller difference seen up to 10 degrees from the western boundary. The sign of this difference is opposite that found in the standard and coarse cases in this region. Its magnitude is dependent on the order of interpolation used and the type of the interface.

Another view of the data will make comparison easier. For all cases, the zonal root-mean-square difference is calculated by

$$\Delta_j^{\mathrm{rms}} = \left[\frac{1}{n_i} \sum_{i'=1}^{n_i} \left(\eta_{i',j} - \eta^f_{i',j} \right)^2 \right]^{\frac{1}{2}}, \tag{4.1}$$

where n_i is the number of cells in the zonal direction, η is the surface height, and η^f is the surface height of the fine resolution case. This quantity is plotted for the various resolutions in Figure 4.14 with the reduced case shown being that with linear interpolation. This more clearly shows the similarity of the reduced grid solution to that of the coarse grid

in the northern half and to the standard grid in the southern half. There is a range within 3 degrees of the interface where the lower differences from the southern region are matched to the higher differences in the northern region. At the higher end of this range, noise can be seen in the plot. This same noise is seen in the two-dimensional plot of Figure 4.13, indicating numerical noise added by the interface. This is not the same numerical noise as discussed in Section 2.3.8, which was a two-dimensional "checkerboarding", though the present noise could be a source of such a component in the solution.

The zonal r.m.s. differences for the various reduced grid interface conditions are shown in Figure 4.15 for a type-1 interface and Figure 4.16 for a type-2 interface. Differences between the reduced grid cases begin to emerge. All of the cases show the oscillatory noise north of the interface that was mentioned previously, with the exception of the type-2 case with no interpolation. But while it doesn't produce this type of noise, the relative size of the error is comparatively large in the vicinity of the interface. The type-1 case with no interpolation has much larger errors than all of the other cases across the whole domain. The type-1 case with linear interpolation shows the smallest error in the vicinity of the interface, as well as the smallest overall error.

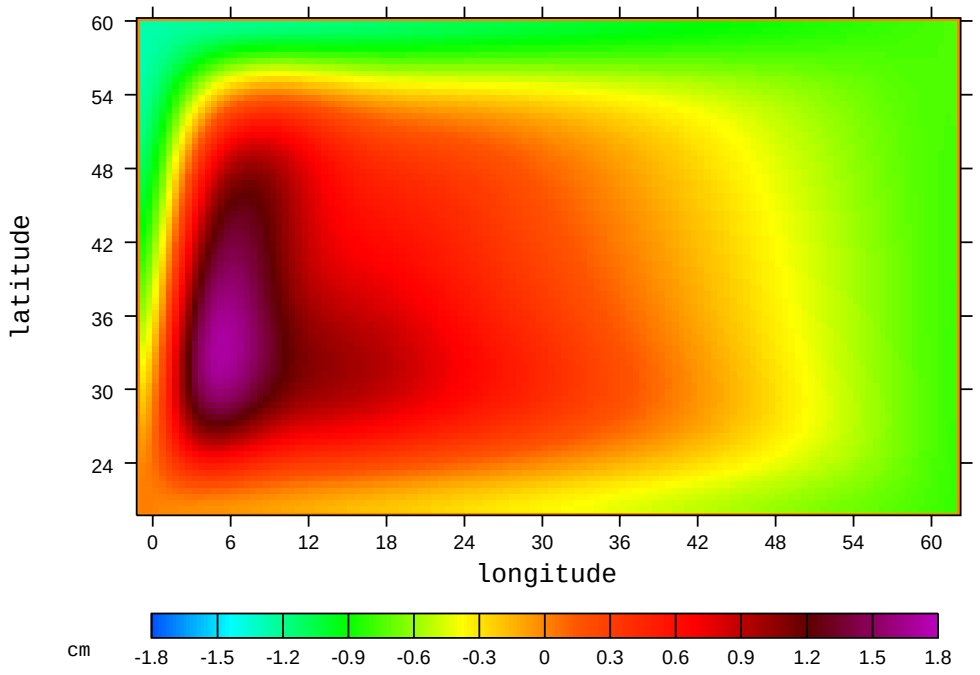

Figure 4.7: Surface height of the linear-wind-forced, fine resolution basin.

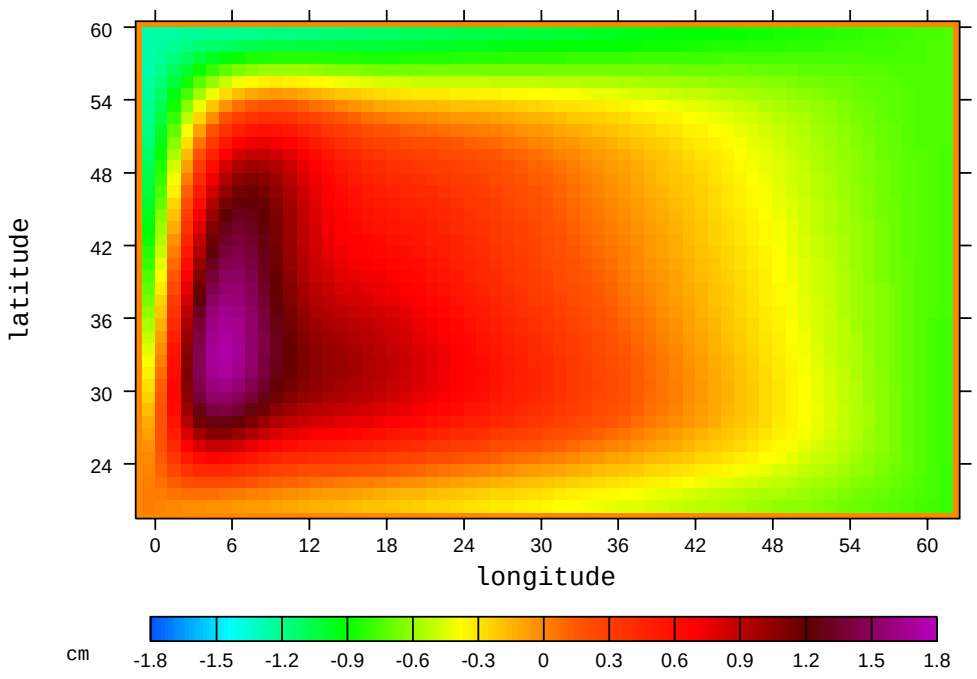

Figure 4.8: Surface height of the linear-wind-forced, standard resolution basin.

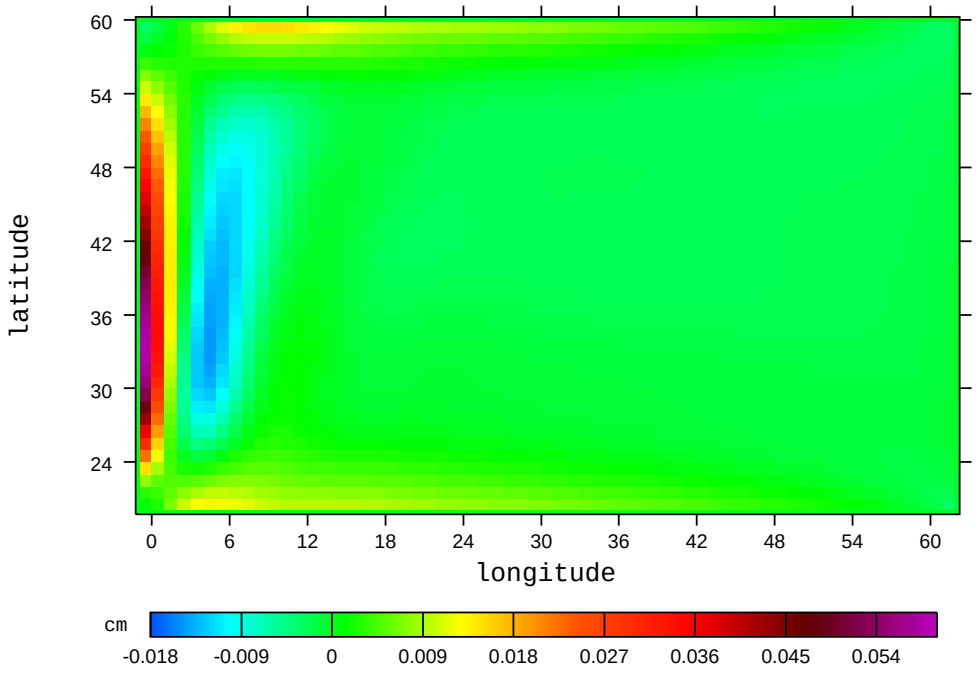

Figure 4.9: Difference between the surface height of the standard resolution and the fine resolution cases of the linear-wind-forced basin.

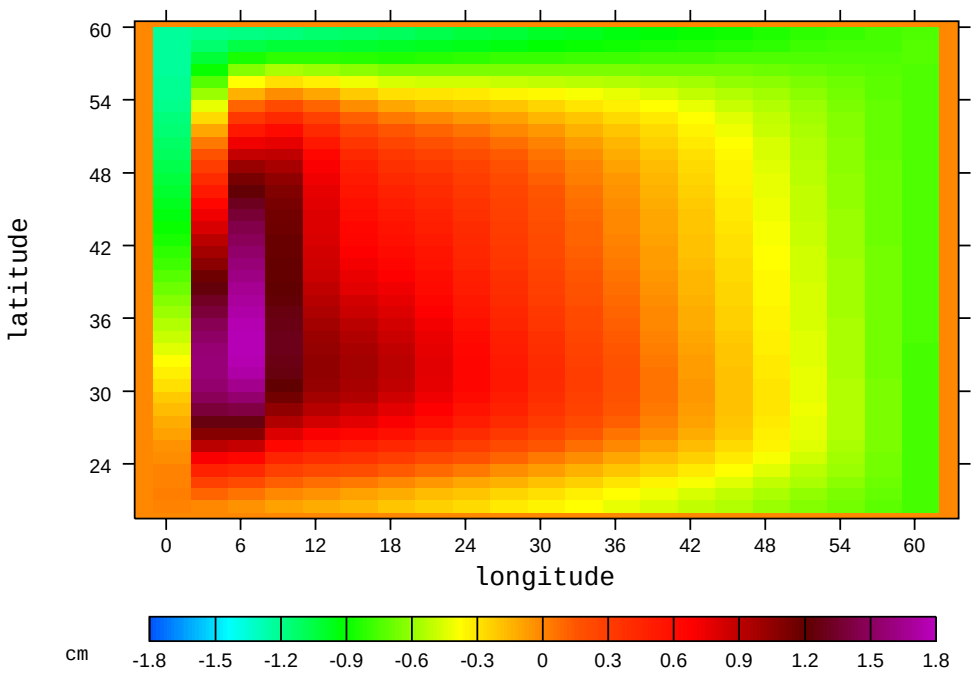

Figure 4.10: Surface height of the linear-wind-forced, coarse resolution basin.

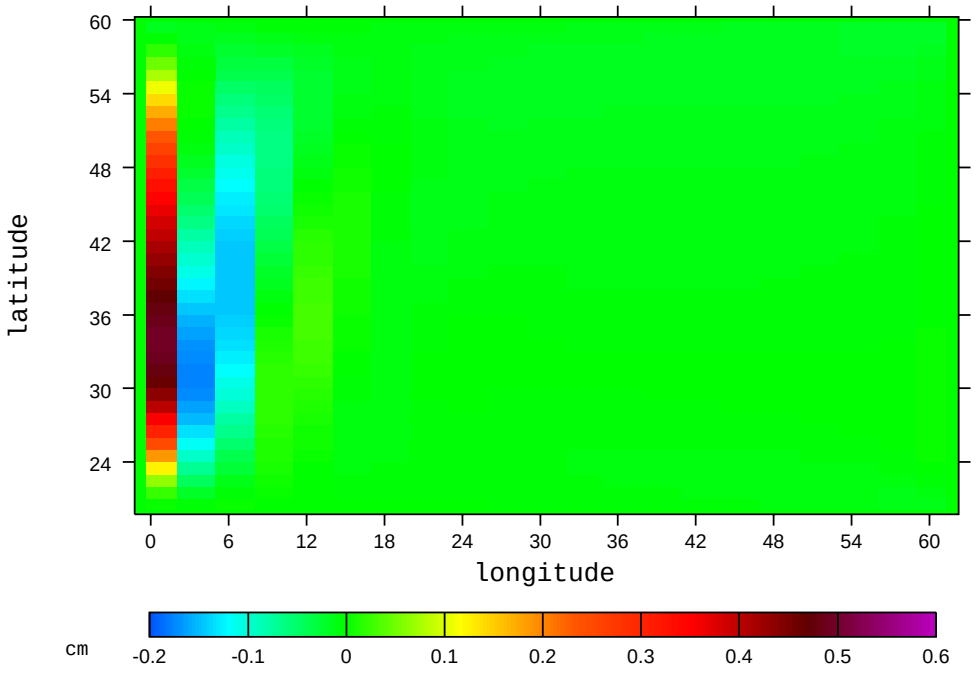

Figure 4.11: Difference between the surface height of the coarse resolution and the fine resolution cases of the linear-wind-forced basin.

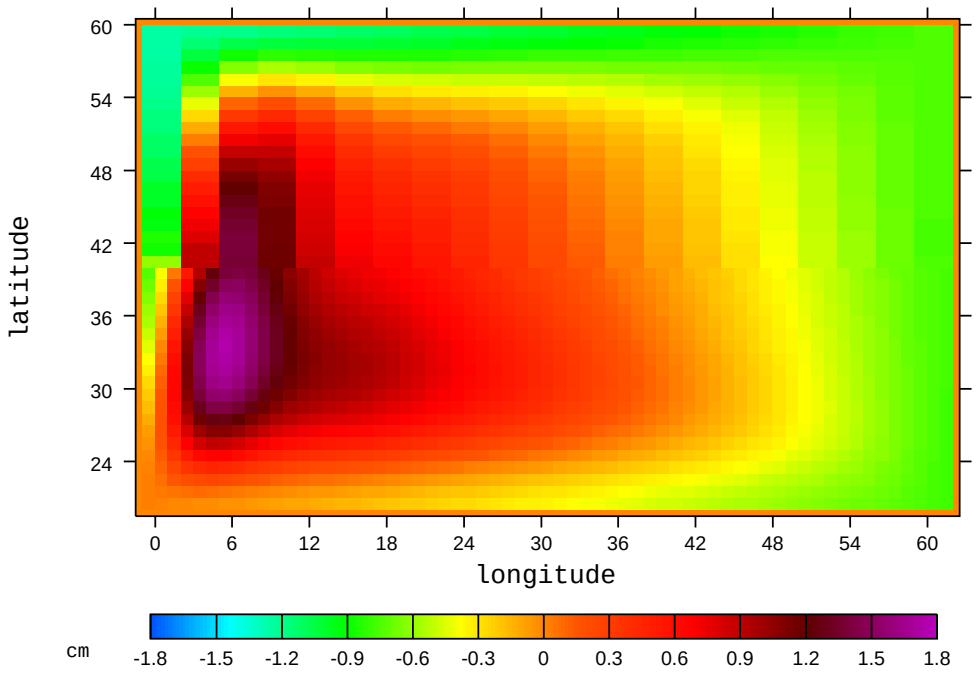

Figure 4.12: Surface height of the linear-wind-forced, type-1 reduced grid basin with linear interpolation at the interface.

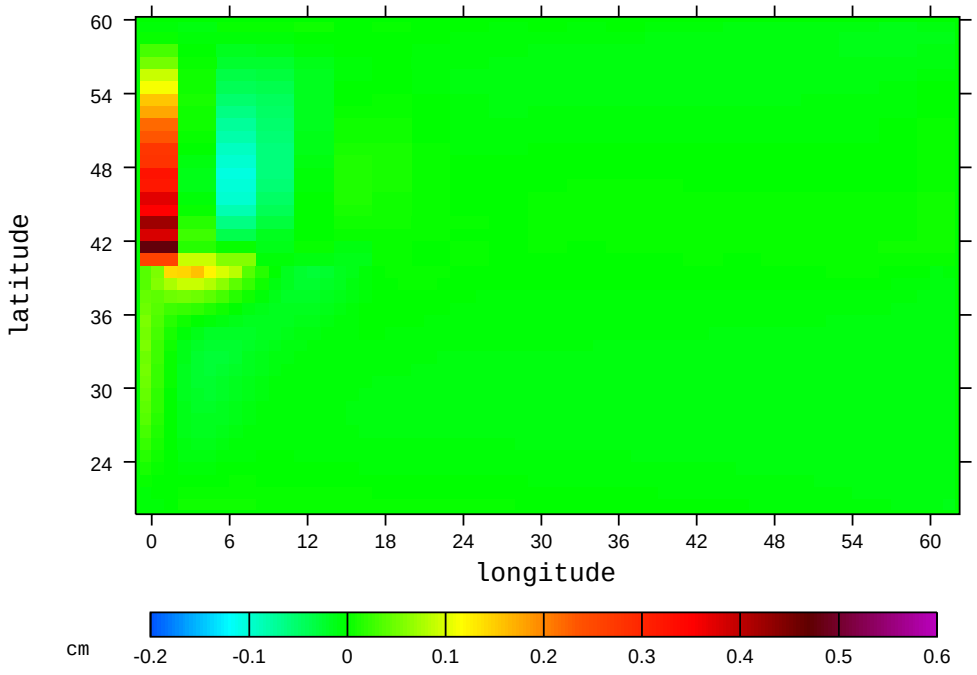

Figure 4.13: Difference between the surface height of the reduced grid and the fine resolution cases of the linear-wind-forced basin, with linear interpolation at the type-1 reduced grid interface.

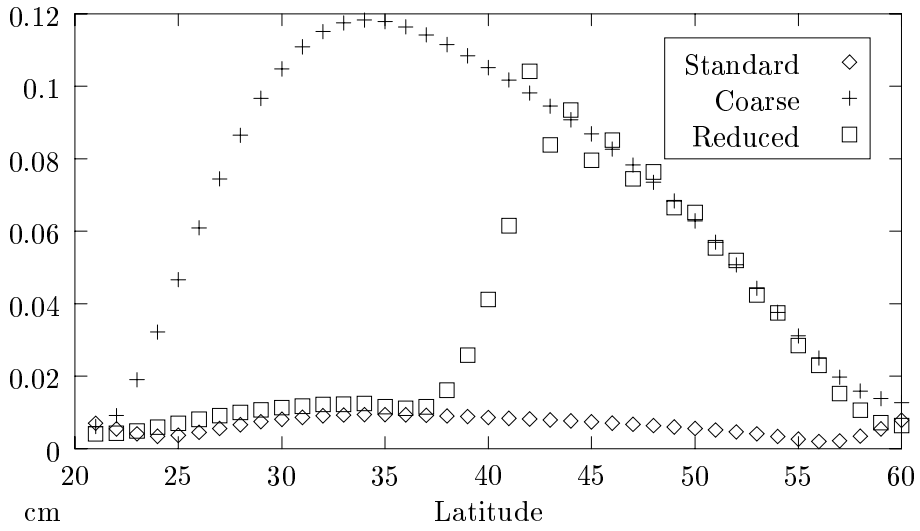

Figure 4.14: Zonal r.m.s. difference in surface height between various grid resolutions and the fine resolution case for the linear-wind-forced basin.

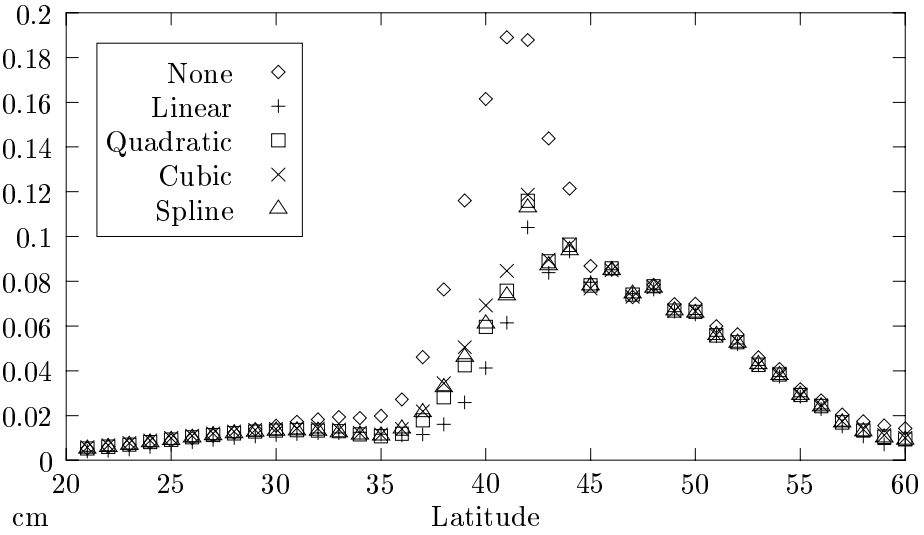

Figure 4.15: Zonal r.m.s. difference in surface height between type-1 reduced grid cases and the fine resolution case for the linear-wind-forced basin by interpolation type

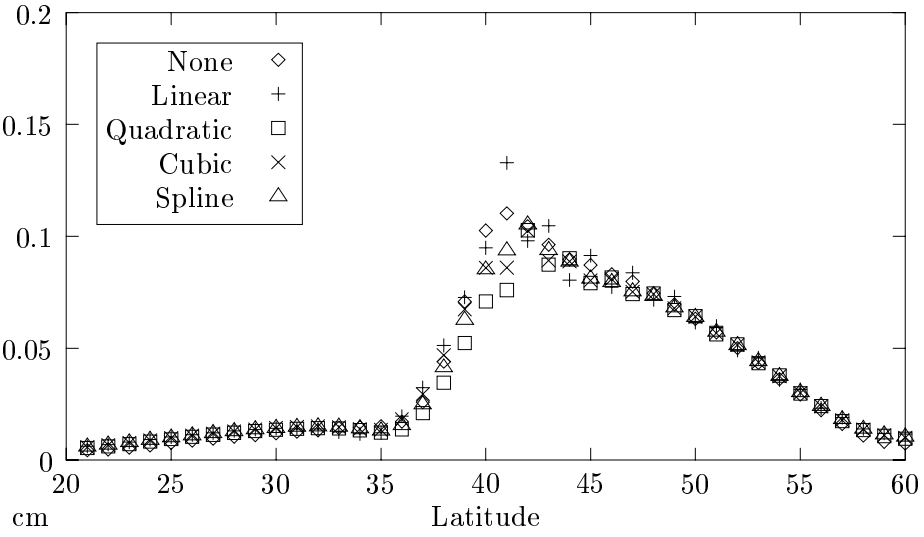

Figure 4.16: Same as Figure 4.15 except for a type-2 interface.

4.1.3 Barotropic Velocity

It is expected that the noise seen in the surface height will occur in the same regions as noise in the velocity. Figure 4.17 shows the velocity for the standard grid case. Once again the narrowed and intensified western boundary is observed. Flow in this boundary current is strongly meridional throughout most of the middle latitudes and thus across any reduced grid interface. Figure 4.18 and Figure 4.19 show the same field for the coarse grid and a representative reduced grid.

Figure 4.20 shows the meridional velocity at 40 degrees latitude, i.e. at the tracer grid interface, for the coarse and standard cases as well as one representative reduced grid case. The fine case is very nearly the same as the standard case and is not shown for clarity. The same data is shown in a different manner in Figure 4.21, where it is integrated from the western boundary to give a northward transport. The coarse grid velocity is seen to be slightly high in the western boundary current with a correspondingly high transport. The representative reduced grid velocity is just the opposite, with transport that is slightly low in the boundary current.

Figure 4.22 and Figure 4.23 show the same data for various types of reduced grid interpolations at type-1 and type-2 interfaces, respectively. For the type-2 interface, however, the velocity interface location has changed, so the data used for these plots is taken from 39 degrees latitude. This makes direct comparison between the type-1 and type-2 interfaces difficult, but each plot shows the standard resolution data at the proper latitude for comparison. Figure 4.24, and Figure 4.25 show the integrated transport for the same cases. Velocities for the type-2 interface are generally similar to one another, again with the exception of the case with no interpolation. There is more variation between the

type-1 interface cases, and the linear interpolation case again stands out as the overall closest to the standard case.

Proceeding to the zonal velocity, Figure 4.26 shows the zonal velocity at 40 degrees latitude, i.e. at the tracer grid interface, for the coarse and standard cases as well as one representative reduced grid case. The fine case is very nearly the same as the standard case and is not shown for clarity. Immediately, the error in the reduced grid velocity stands out, well larger than the coarse grid velocity error. Figure 4.27 and Figure 4.28 show the same data for the various types of reduced grid interpolations and interface types. All of them have the same large zonal velocity at the western edge, generally with a fair amount of noise. The type-1 cases with no interpolation and with cubic spline interpolation are noteworthy for their large oscillations. The type-2 case with no interpolation gives results nearly as poor as the comparable type-1 case. The other cases are similar to one another. The only stand-out is the type-2 case with linear interpolation, which is somewhat more smooth and has dropped to near the standard values by 10 degrees longitude. Results from another wind-driven basin in which there is more zonal velocity along the interface will be presented later to determine whether or not the error grows with the strength of the flow.

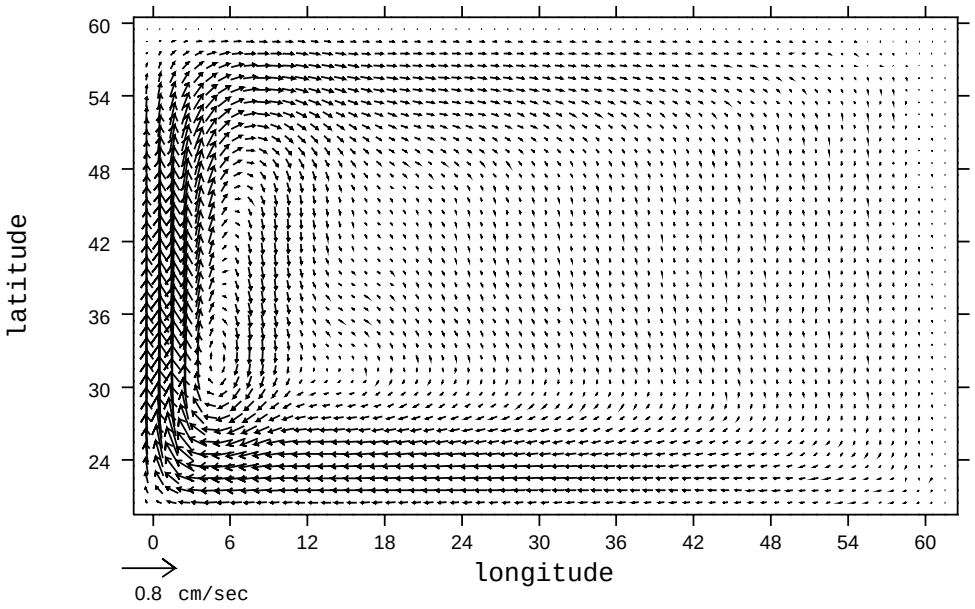

Figure 4.17: Barotropic velocity of the linear-wind-forced, standard resolution basin.

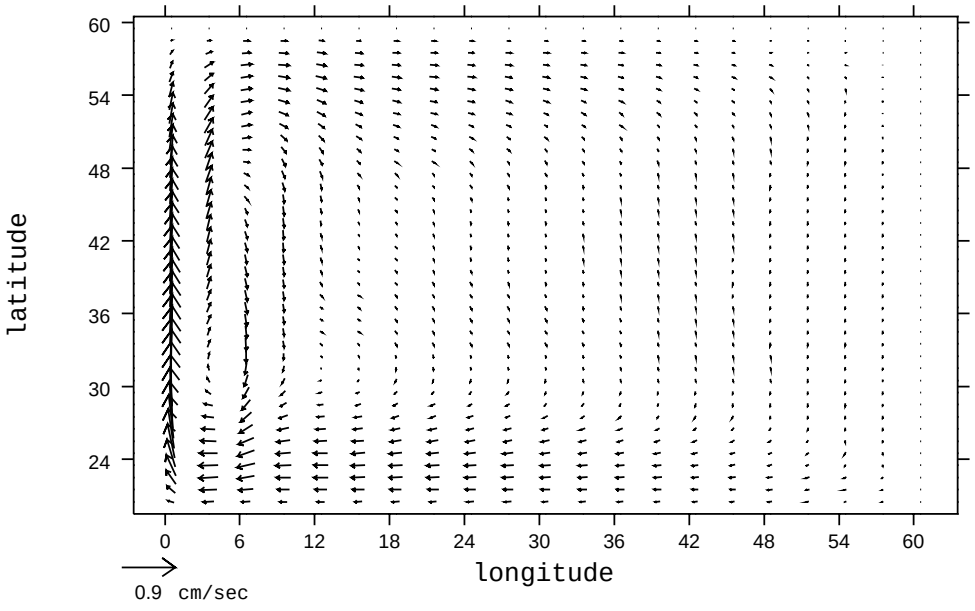

Figure 4.18: Barotropic velocity of the linear-wind-forced, coarse resolution basin.

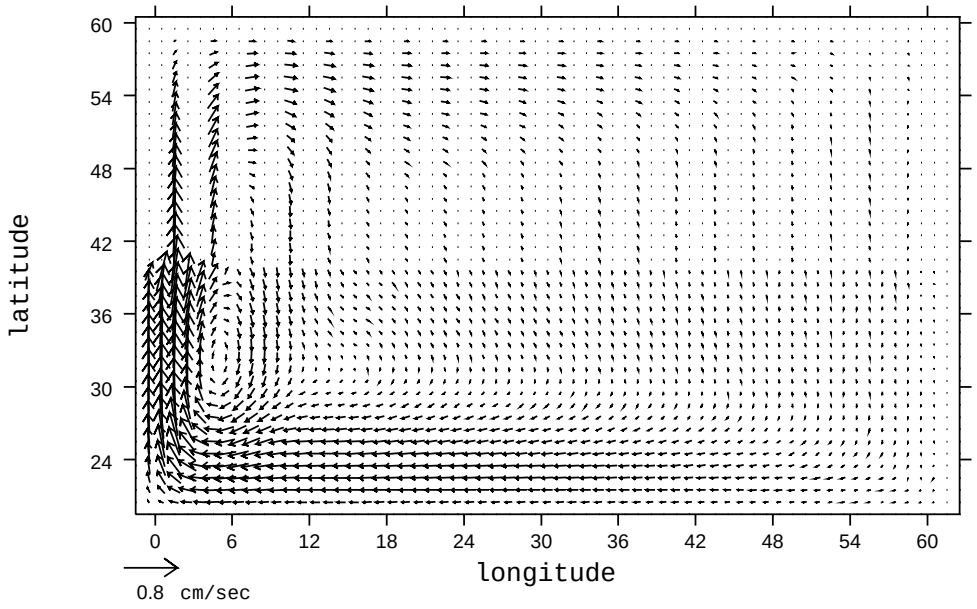

Figure 4.19: Barotropic velocity of the linear-wind-forced, reduced grid basin.

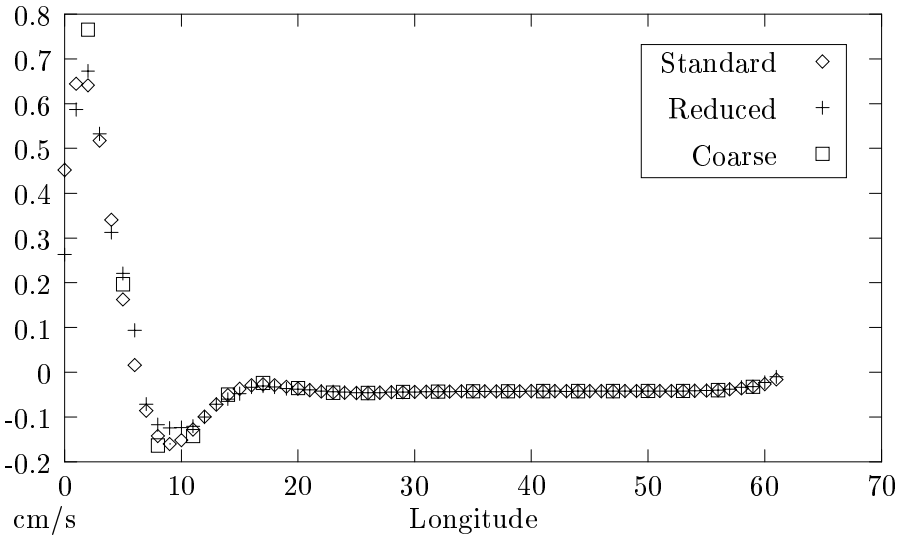

Figure 4.20: Meridional barotropic velocity of the linear-wind-forced basin at 40 degrees latitude by grid resolution.

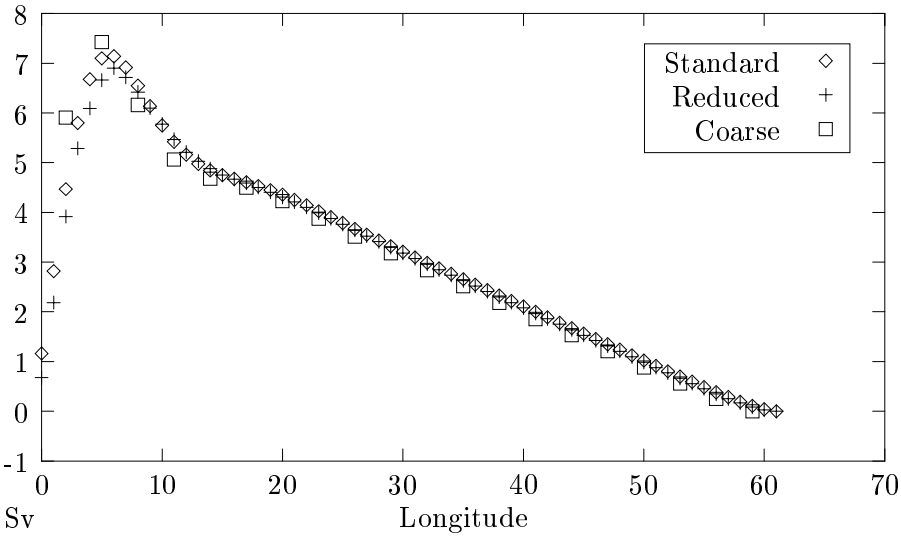

Figure 4.21: Integrated northward transport of the linear-wind-forced basin at 40 degrees latitude by grid resolution.

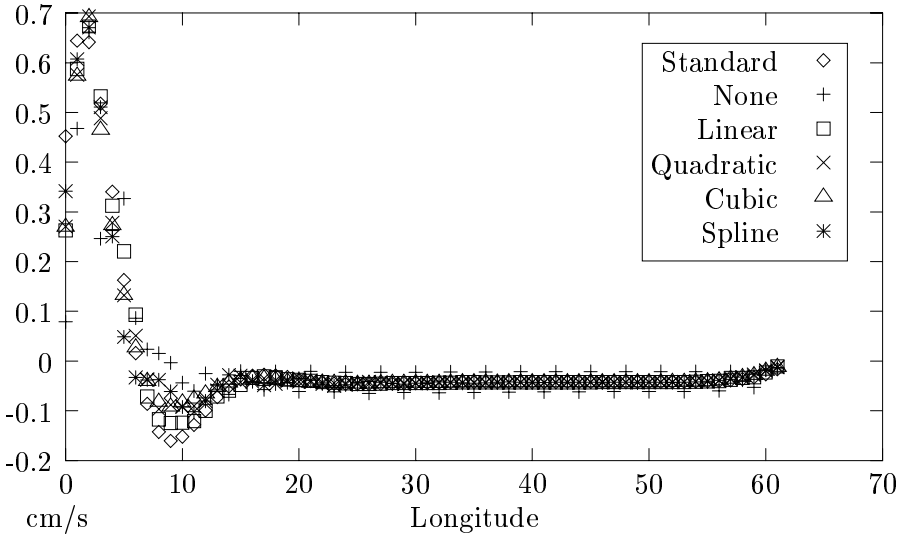

Figure 4.22: Meridional barotropic velocity of the linear-wind-forced basin at the type-1 reduced grid interface by interpolation type. Results from the standard resolution case are shown for comparison.

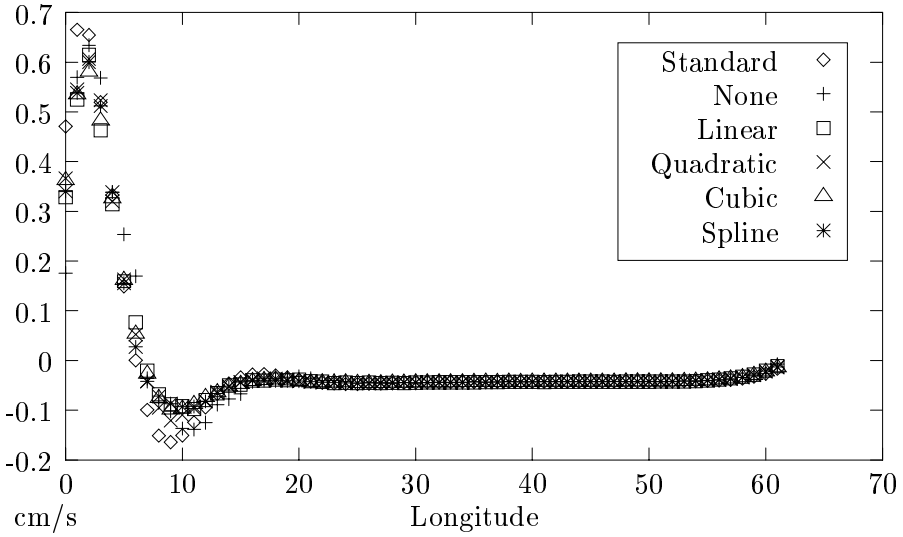

Figure 4.23: Same as Figure 4.22 except for a type-2 interface.

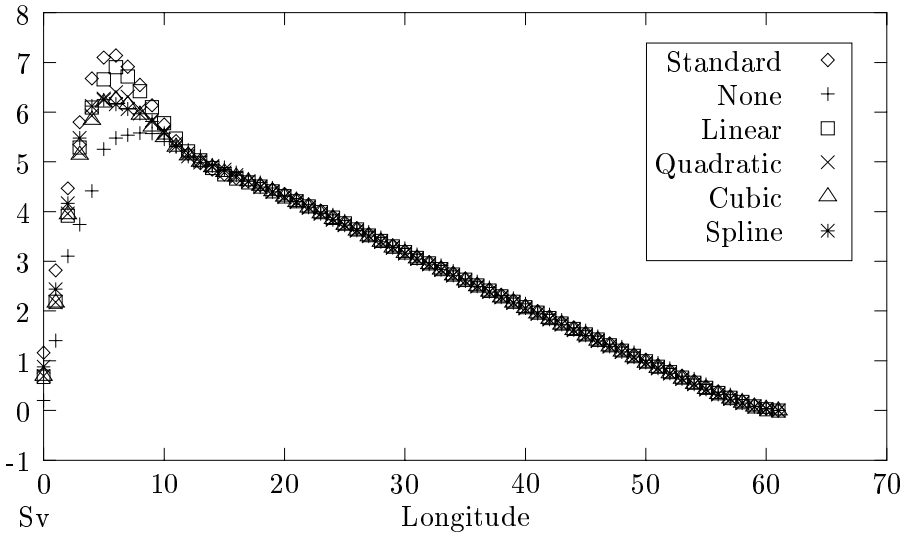

Figure 4.24: Integrated northward transport of the linear-wind-forced basin at the type-1 reduced grid interface by interpolation type. Results from the standard resolution case are shown for comparison.

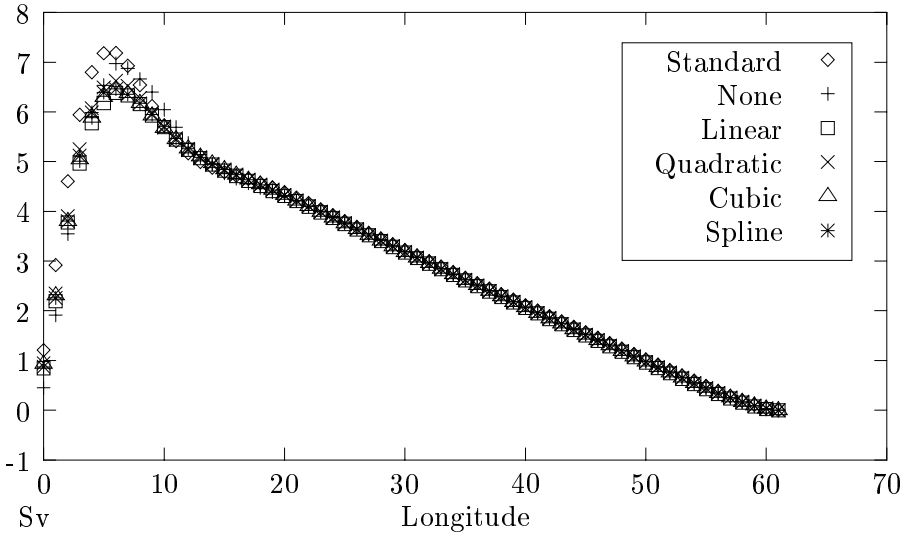

Figure 4.25: Same as Figure 4.24 except for a type-2 interface.

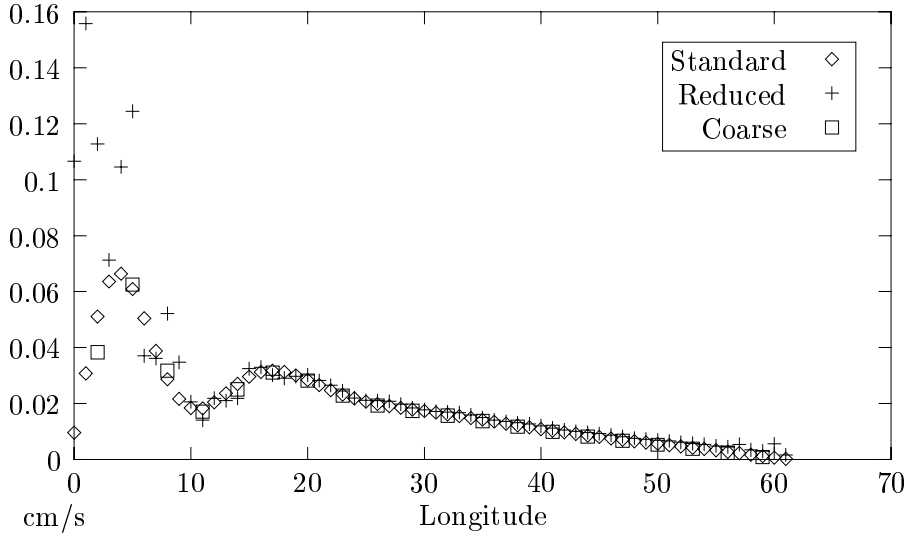

Figure 4.26: Zonal barotropic velocity of the linear-wind-forced basin at 40 degrees latitude by grid resolution.

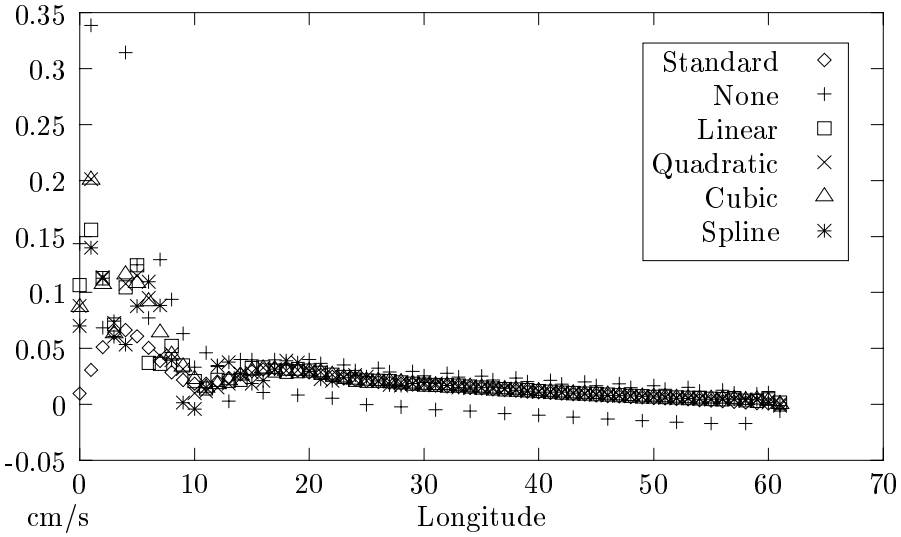

Figure 4.27: Zonal barotropic velocity of the linear-wind-forced basin at the type-1 reduced grid interface by interpolation type. Results from the standard resolution case are shown for comparison.

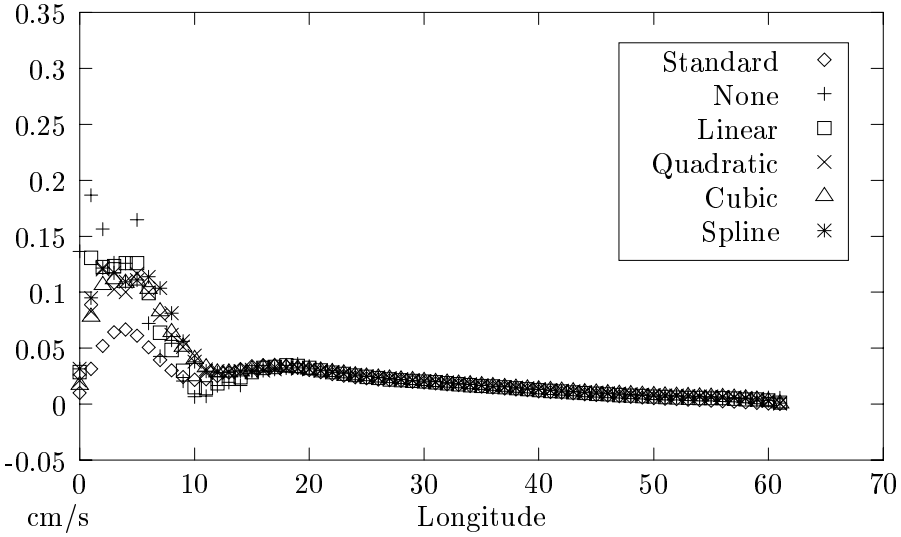

Figure 4.28: Same as Figure 4.27 except for a type-2 interface.

4.1.4 Meridional Overturning

An important baroclinic flow is the meridional overturning, given by the zonally averaged meridional velocity integrated from the bottom to the surface. It is obtained by the formula

$$\Psi_{j,k} = \left(-10^{-6}\right) h \cos\phi_j \sum_{k'=km}^{k} dz_{k'} \sum_i v_{i,j,k'}, \tag{4.2}$$

where h is the grid spacing in meters at the equator, dz_k is the thickness of the level k in meters, and v is the baroclinic meridional velocity in meters per second. The factor of 10^{-6} converts to Sverdrups, or 10^6 m^3/s.

Figure 4.29 shows the meridional overturning for the standard resolution case. The basin has upwelling at the northern and southern boundaries with weaker downwelling throughout most of the domain. The minimum values of the streamfunction occur in the middle latitudes of the basin. Again, the fine and standard resolution streamfunctions are nearly identical, so only one is shown.

The overturning is very similar for all cases, so differences are plotted. Figure 4.30 shows the difference between the coarse and standard resolution overturning. In both the northern and the southern halves of the domain the coarse resolution overturning is too weak by a few percent. Figure 4.31 shows the difference between a reduced grid case and the standard case on the same scale. In the northern half of the domain, where resolution is the same as that of the coarse grid, the differences are nearly identical. However, in the southern half, where the resolution is the same as the standard grid, the errors are very small. This is similar to the behavior found above for other flow properties, but the overturning does not seem to suffer from numerical noise generated by the interface.

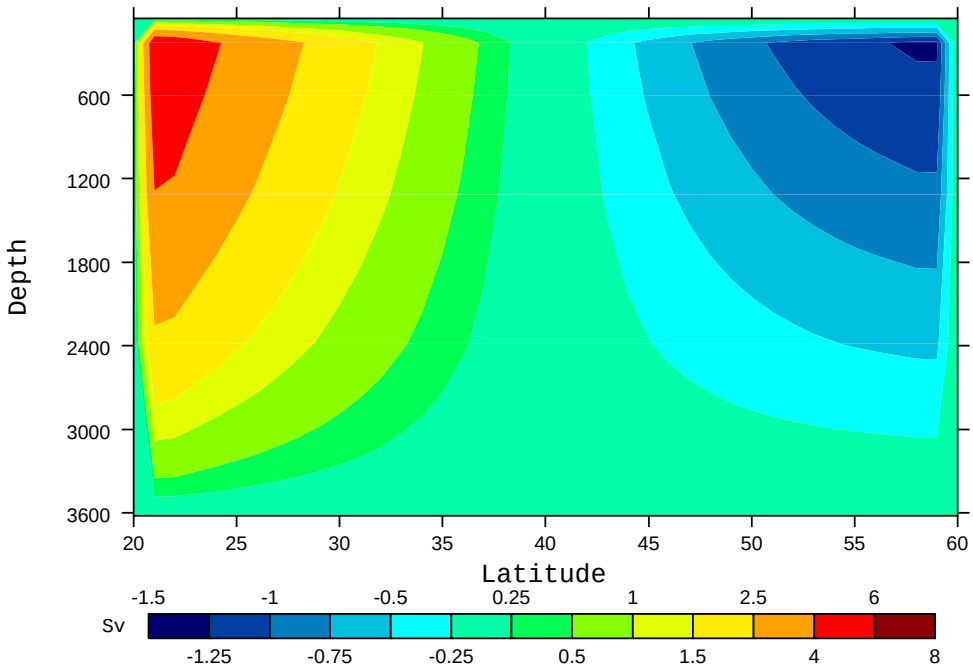

Figure 4.29: Meridional overturning of the linear-wind-forced, standard resolution basin. Positive values indicate clockwise circulation.

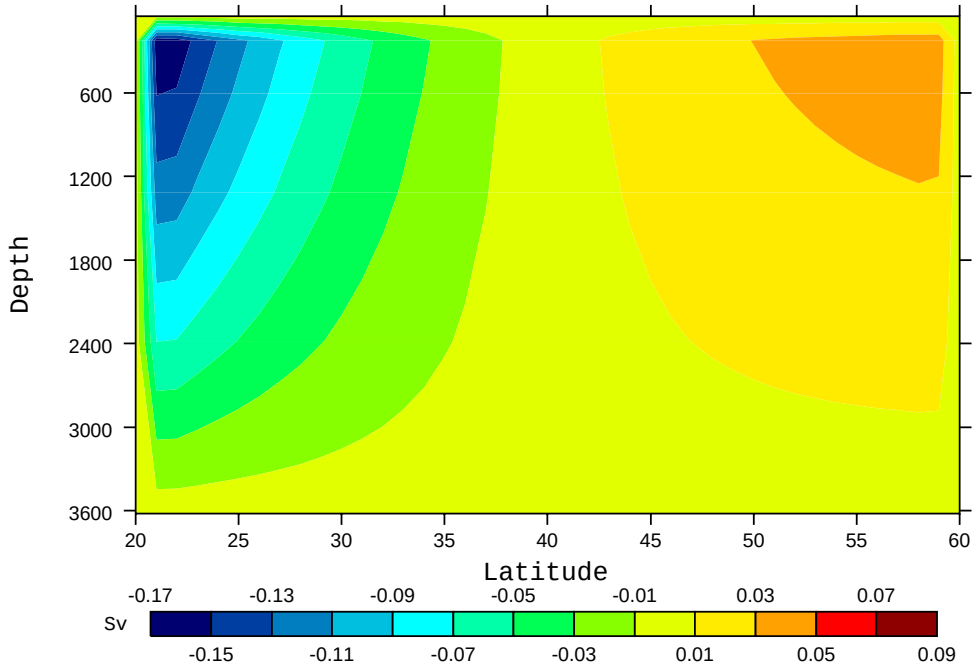

Figure 4.30: Difference between the meridional overturning of the coarse and standard resolution cases of the linear-wind-forced basin.

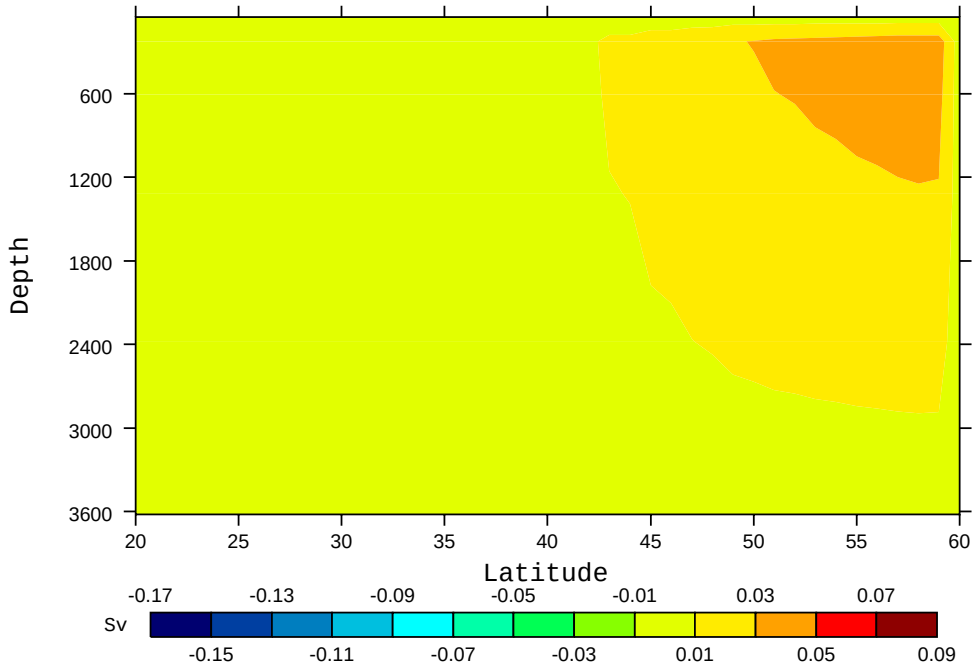

Figure 4.31: Same as Figure 4.30 except for the reduced grid and standard cases.

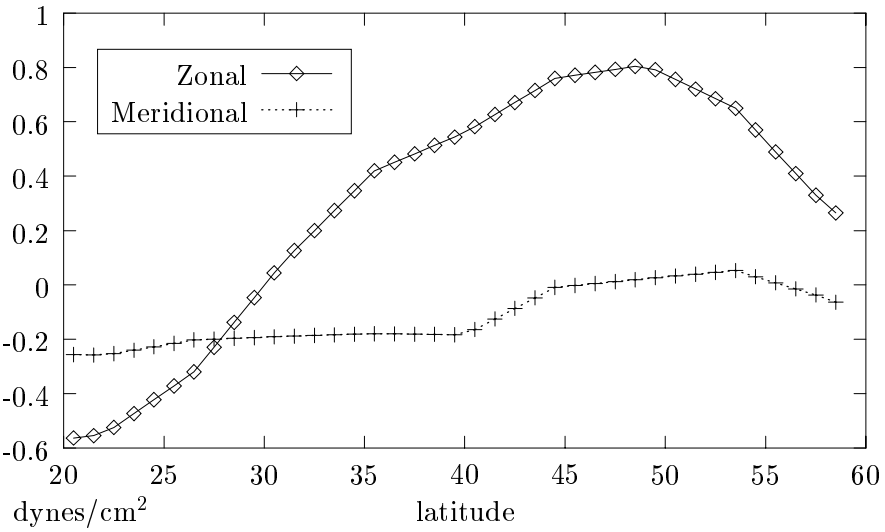

Figure 4.32: Zonal and Meridional surface wind stress for the "real"-wind-forced basin. Both components of the forcing are zonally constant.

4.2 Another Wind-Driven, Flat Bottomed Basin

The previous test had strong flow across the reduce grid interfaces, but had only weak flow along the interfaces. Results from another wind-driven basin will be presented here in which there is greater flow along the interface. The only changes made to the setup of the runs is the wind forcing, which is shown in Figure 4.32. There is now a meridional component to the wind stress, though both the zonal and meridional components are still zonally constant. This wind forcing is based on observed global, annual mean wind stress data from Hellerman and Rosenstein in [32]. Because this forcing is derived from data, I will refer to this test case as the "real"-wind-forced basin.

The results of the previous wind-driven basin have demonstrated that the reduced grid with no interpolation produces large errors at the interfaces. Because of these errors it will not be included in further tests, even though this method was of interest for its conservation properties as detailed in Section A.1. Therefore, fine, standard, and coarse

resolution cases are run with the modified forcing along with reduced grid cases having linear, quadratic, cubic polynomial, and cubic spline interpolation for both type-1 and type-2 interfaces.

4.2.1 Barotropic Velocity

The results for kinetic energy and surface height are very similar to the previous wind-driven case, and all of the comments under that section apply equally to the results of this case. But the velocity at the interface is the reason for this run, and Figure 4.33 shows the barotropic flow field for the standard resolution case. Notice that the flow at 40 degrees latitude has a much stronger zonal component than the previous case. The flow is now a double gyre, though the narrowed and intensified western boundary current remains.

Figure 4.34 shows the meridional velocity at 40 degrees latitude, i.e. at the tracer grid interface, for the coarse and standard cases as well as one representative reduced grid case. The fine case is very nearly the same as the standard case and is not shown for clarity. The same data is shown in a different manner in Figure 4.21, where it is integrated from the western boundary to give a northward transport. The coarse grid velocity is seen to be slightly high in the western boundary current with a correspondingly high transport. The representative reduced grid velocity is just the opposite, with transport that is slightly low in the boundary current.

Figure 4.36 and Figure 4.37 show the same data for various types of reduced grid interpolations at type-1 and type-2 interfaces, respectively. Again, the type-1 data is from 40 degrees latitude, while the type-2 data is from 39 degrees, with the appropriate standard case values shown for comparison. Figure 4.38, and Figure 4.39 show the integrated

transport for the same cases. Again the velocities for the type-2 interface are generally similar to one another. The best of these is the case with quadratic interpolation. There is more variation between the type-1 interface cases, and the linear interpolation case again stands out as the overall best.

Figure 4.40 shows the zonal velocity at 40 degrees latitude, i.e. at the tracer grid interface, for the coarse and standard cases as well as the type-1 reduced grid case with linear interpolation. The fine case is very nearly the same as the standard case and is not shown for clarity. Compared to the linear-wind-forced basin, the reduced grid velocity is much closer to the standard result, though it is again too high in the boundary current region.

Looking more closely at the reduced grid results, Figure 4.41 and Figure 4.42 show the same data for the various types of reduced grid interpolations and interface types. All have a zonal velocity which is too large at the western edge of the domain, though they are smoother with this forcing than with the linear zonal forcing. The linear case is again the closest, though it is perhaps the least smooth.

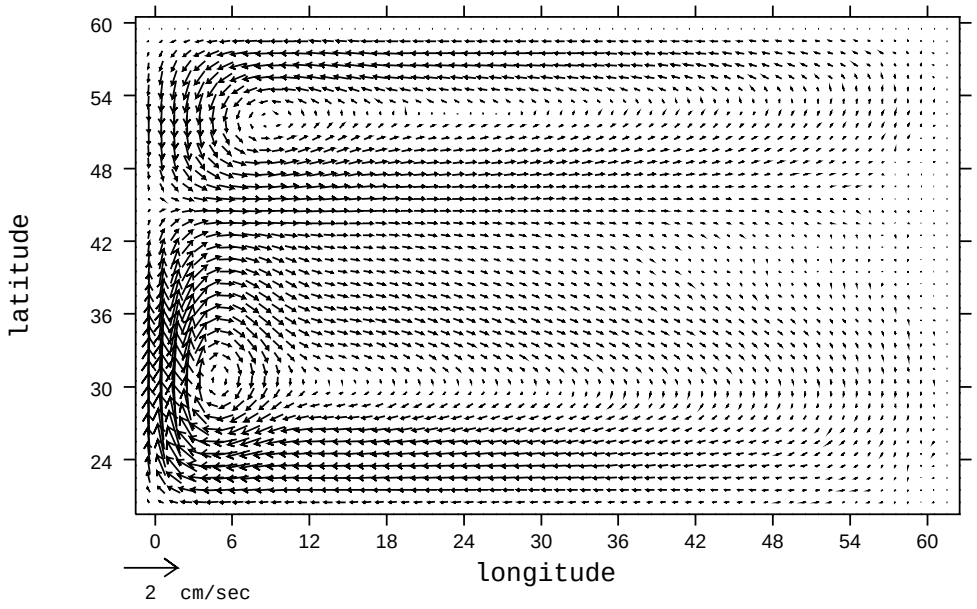

Figure 4.33: Barotropic velocity of the "real"-wind-forced, standard resolution basin.

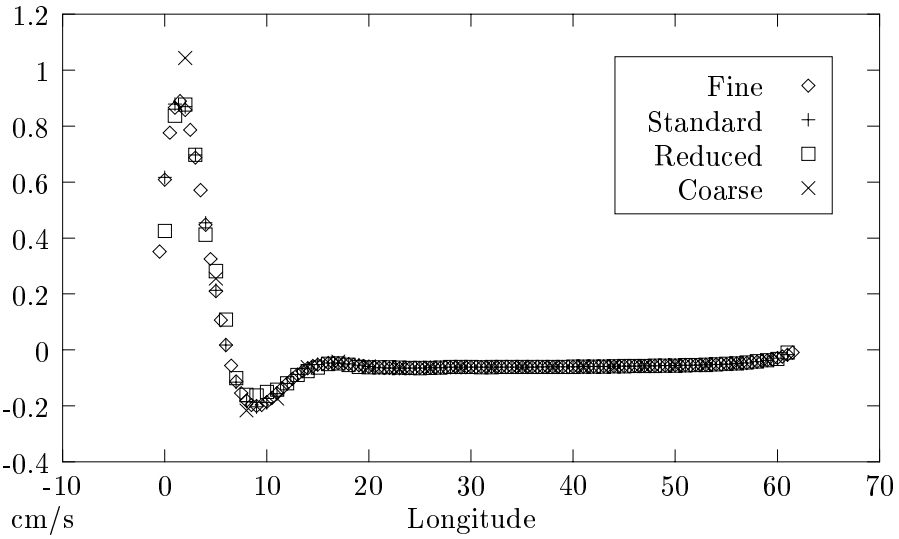

Figure 4.34: Meridional barotropic velocity of the "real"-wind-forced basin at 40 degrees latitude by grid resolution.

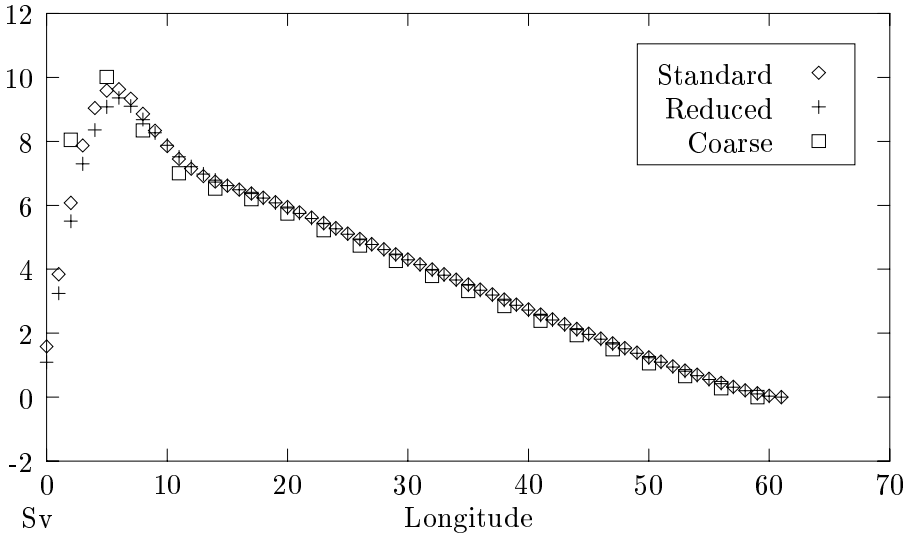

Figure 4.35: Integrated northward transport of the "real"-wind-forced basin at 40 degrees latitude by grid resolution.

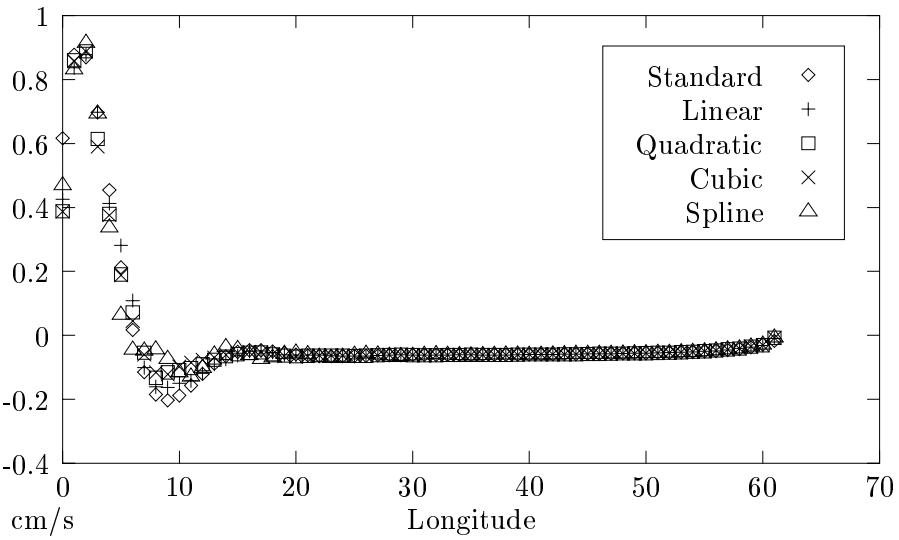

Figure 4.36: Meridional barotropic velocity of the "real"-wind-forced basin at the type-1 reduced grid interface by interpolation type.

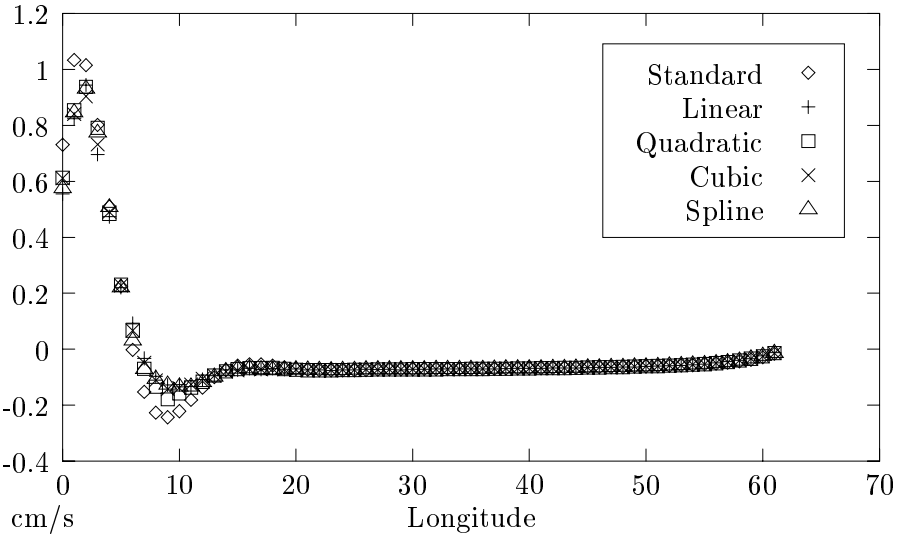

Figure 4.37: Same as Figure 4.36 except for a type-2 interface.

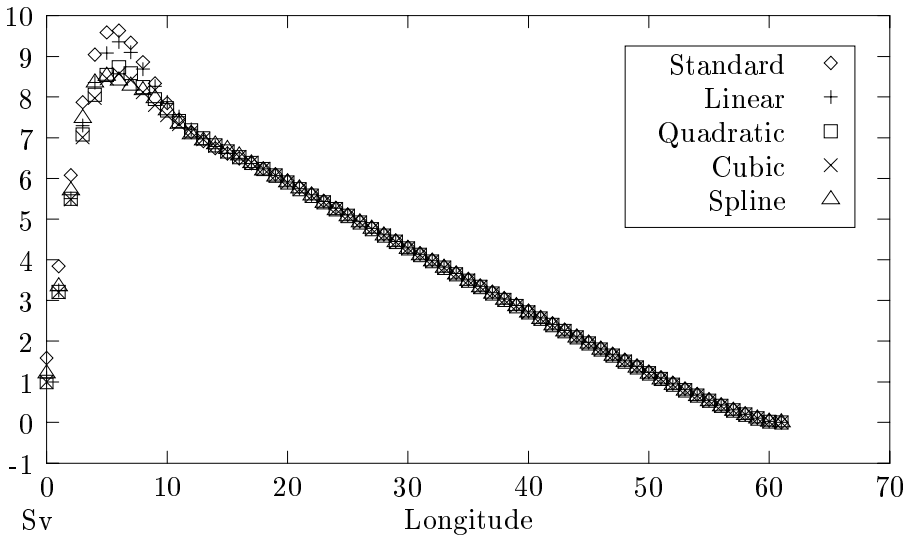

Figure 4.38: Integrated northward transport of the "real"-wind-forced basin at the type-1 reduced grid interface by interpolation type.

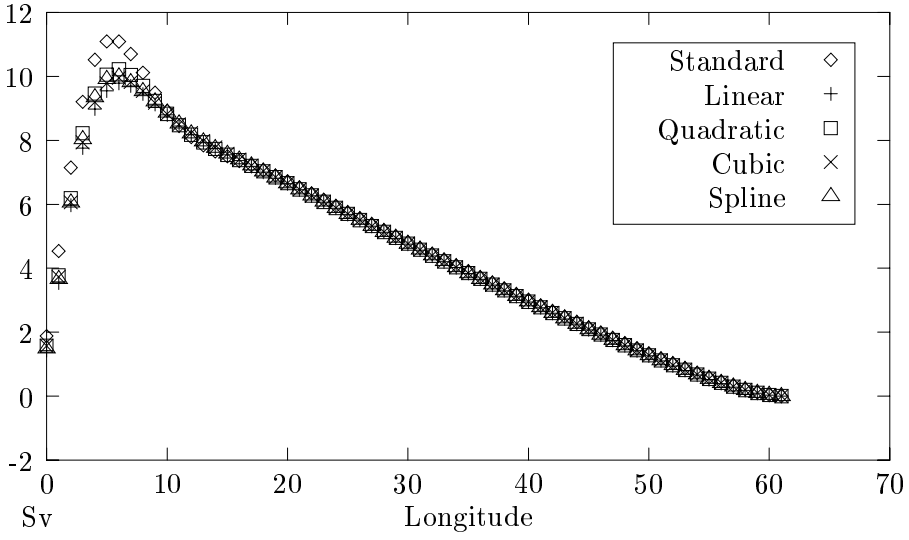

Figure 4.39: Same as Figure 4.38 except for a type-2 interface.

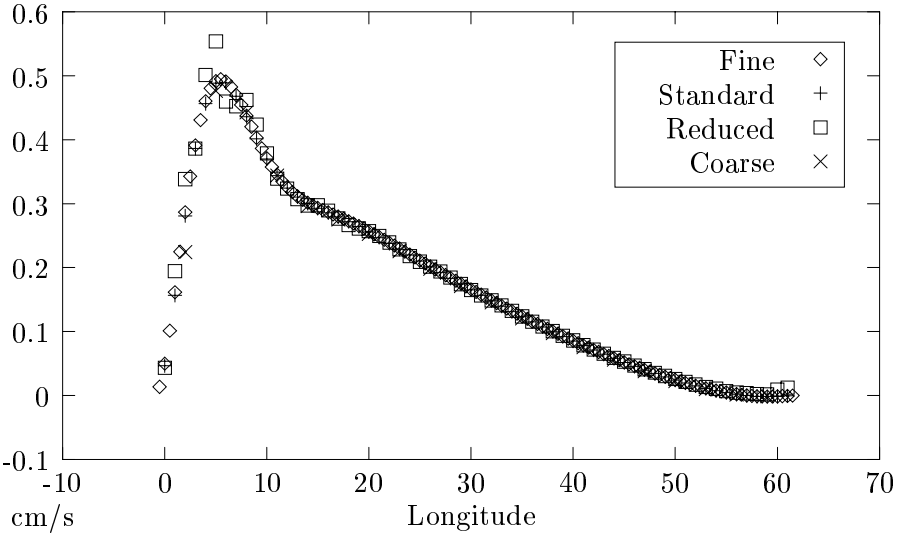

Figure 4.40: Zonal barotropic velocity of the "real"-wind-forced basin at 40 degrees latitude by grid resolution.

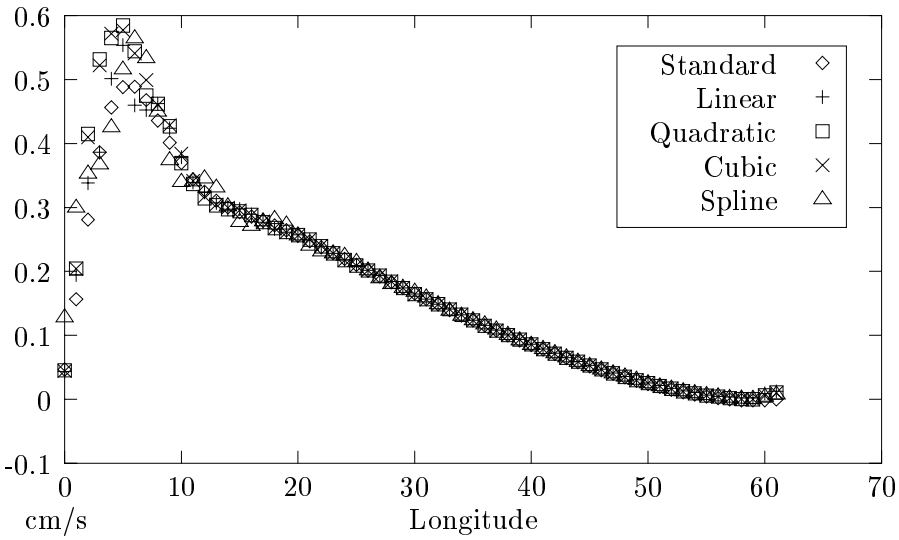

Figure 4.41: Zonal barotropic velocity of the "real"-wind-forced basin at the type-1 reduced grid interface by interpolation type.

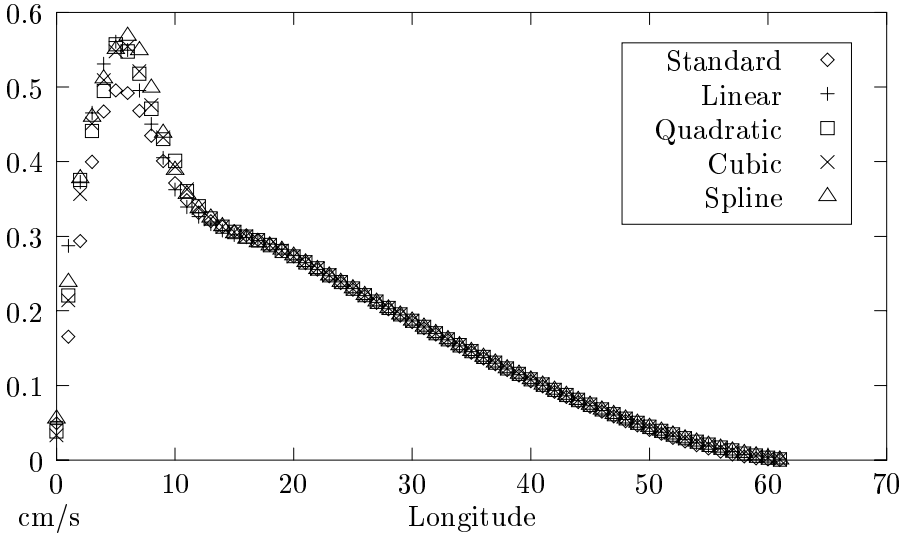

Figure 4.42: Same as Figure 4.41 except for a type-2 interface.

4.2.2 Meridional Overturning

Figure 4.43 shows the meridional overturning for the standard resolution case. Again the basin has upwelling at the northern and southern boundaries with weaker downwelling throughout most of the domain. However, this time the region of maximum downwelling is shifted further south.

Figure 4.44 shows the difference between the coarse and standard resolution overturning. As in the linear-wind-forced basin, the overturning is too weak by a few percent throughout the domain. The difference between the reduced grid and the standard grid, shown in Figure 4.45, is even more striking with the current forcing. In the previous basin, the grid interface was located in a region where the value of the overturning was small and relatively constant. This time the values are larger in the neighborhood of the interface, yet the difference again drops to nearly zero immediately on crossing the interface to the southern half of the basin. However, some numerical noise is generated, as the streamlines are less smooth.

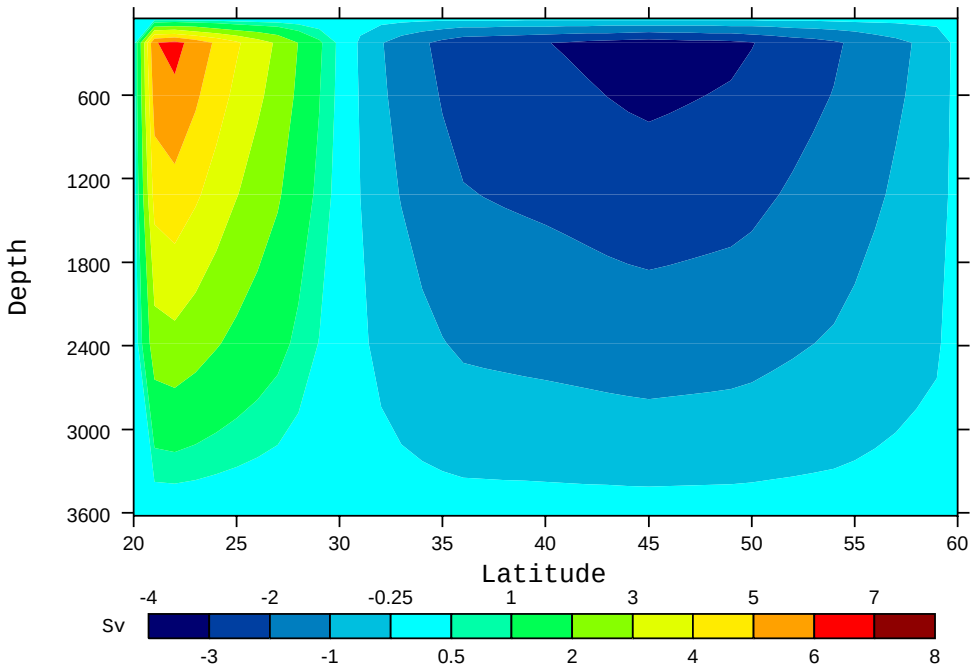

Figure 4.43: Meridional overturning of the "real"-wind-forced, standard resolution basin. Positive values indicate clockwise circulation.

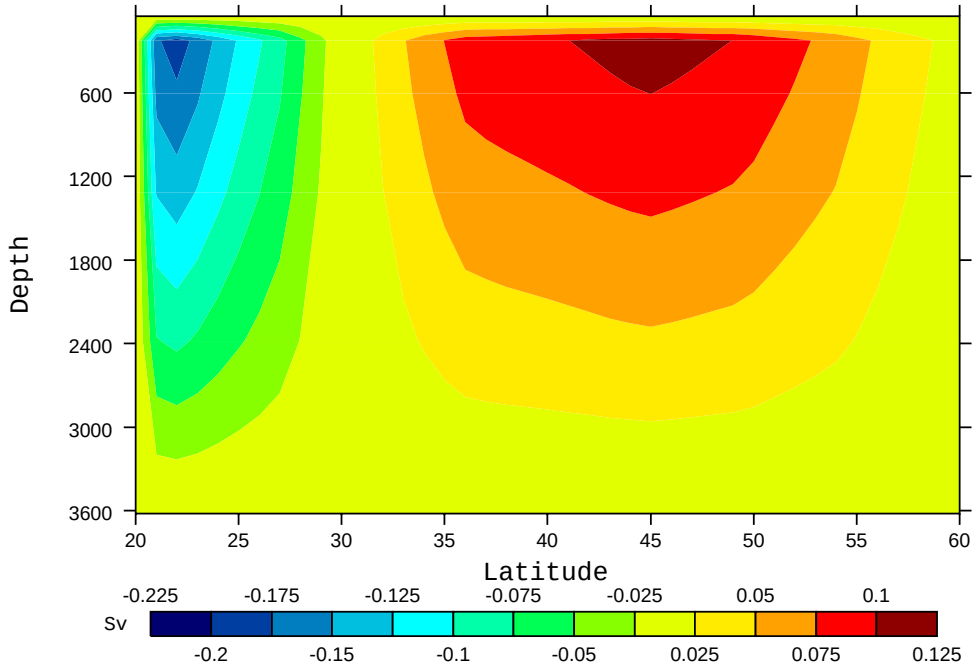

Figure 4.44: Difference between the meridional overturning of the coarse and standard resolution cases of the "real"-wind-forced basin.

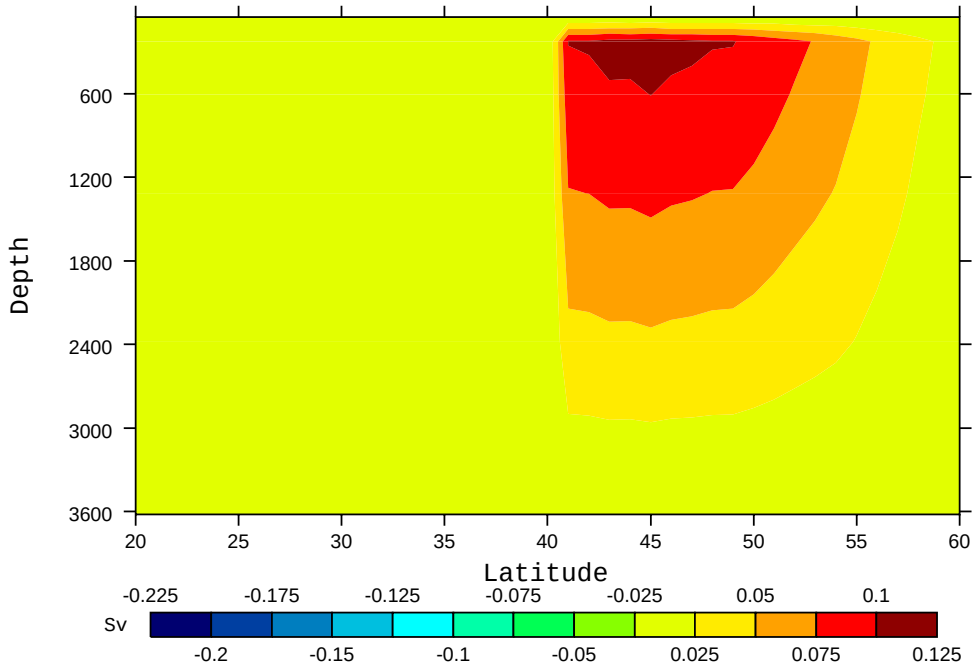

Figure 4.45: Same as Figure 4.44 except for the reduced grid and standard cases.

4.3 Wind and Thermohaline Driven, Flat Bottomed Basin

Now that some of the basic features of the reduced grid model have been explored, longer runs can be made. Whereas in the previous basin runs the time necessary to bring the model to a steady-state was reduced by eliminating thermohaline forcing, the basin runs described in this section will use both wind and thermohaline forcing. The time it takes the system to reach a steady-state solution is very much longer in this case. Whereas the wind-forcing only cases took only 150 simulated days to reach a quasi-steady-state, it takes thousands of years for the current case to do so.

Because of the much larger expenditure of computer time to produce these results, not as many runs will be made. The results of the wind-driven basins have provided enough information to narrow this case to three runs. Once again a standard resolution case, having grid spacings of 1 degree in both longitude and latitude, and a coarse resolution case, having grid spacings of 3 degrees longitude and 1 degree latitude, will be used for comparison against the reduced grid. The reduced grid method which consistently produced the best results in the previous two tests was that with a type-1 interface and linear interpolation. This will be the only method used for the remaining model tests.

The domain will still be the flat-bottomed basin described in Section 4.1, and the wind stress forcing will be the forcing of Figure 4.32. The added thermohaline forcing consists of Newtonian surface restoring flux given by

$$\text{Flux}_T = -\frac{dzt_{k=1}}{\tau}\left(T_{k=1} - T^*\right), \tag{4.3}$$

where $dzt_{k=1}$ is the thickness of the top layer, τ is a restoring timescale, and T^* is the reference surface value toward which to restore. In all of these runs, a value of 30 days is used for τ. The reference surface values used for the basin are zonally constant with a

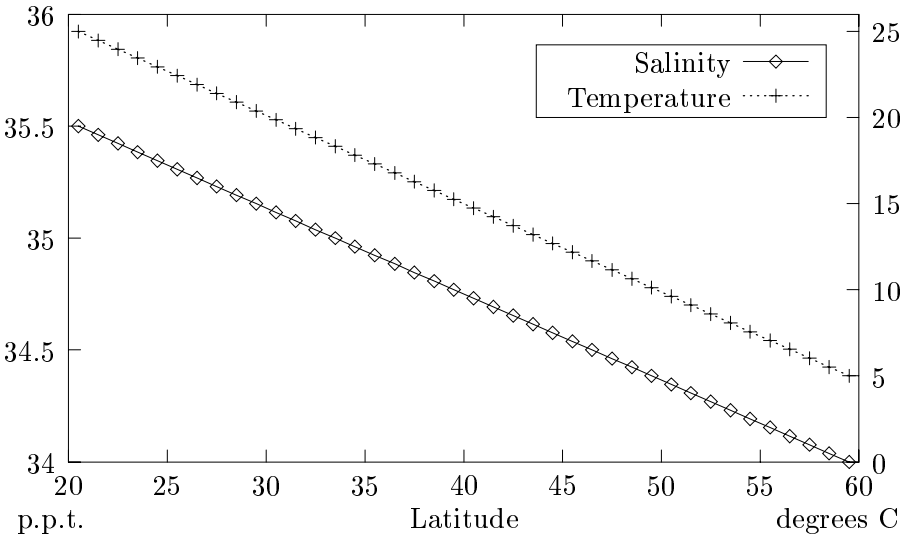

Figure 4.46: The reference surface values of temperature and salinity to use in calculating surface thermohaline forcing for the fully-forced basin.

linear profile, shown in Figure 4.46. Both the wind and thermohaline forcings are time-independent. The horizontal tracer diffusion coefficient is 2×10^7 cm^2, while the vertical tracer diffusion coefficient varies with depth, from 0.2 cm^2/s in the top layer to 1.3 cm^2/s for the bottom layer. The initial condition for temperature is a highly idealized, zonally constant depth profile producing a value of 2.0 degrees below 2000 meters. Salinity is initialized to a constant 34.9 parts per thousand (ppt).

Timesteps for the spinup of the model to equilibrium are 1 day for the tracers and $\frac{1}{2}$ hour for the momentum. The use of unequal timesteps to accelerate the convergence of ocean models to a steady-state is discussed in [17]. The barotropic equations are subcycled 40 times per baroclinic timestep. Each case was run with these steps for at least 8000 tracer years, bringing the solutions close to equilibrium. At the end of this initial run, each was run for an additional 10 years with equal timesteps of $\frac{1}{2}$ hour to eliminate any errors associated with time step splitting.

Grid	K. E. dynes/cm^3	% "error"
Coarse	0.3899	3.7
Reduced	0.3686	2.0
Standard	0.3761	n/a

Table 4.5: Steady-state kinetic energy of the wind and thermohaline driven basin for the three grid resolution cases.

4.3.1 Barotropic Velocity

The final kinetic energies of the three cases are shown in Table 4.5. The kinetic energies are over 25 times larger than in the wind-driven only runs, as there is stronger baroclinic flow brought about by the density forcing. Both the coarse and reduced grid cases are closer to the standard case than in the wind-driven only runs. The barotropic velocities are shown in Figure 4.47 through Figure 4.49 and are very similar to those of the "real"-wind-forced basin of Section 4.2.

The velocities at the interface are shown in Figure 4.50 through Figure 4.52. Referring back to the velocities of the "real"-wind-driven runs (Figure 4.34 to Figure 4.42), it is apparent that the differences in the interface velocities of the reduced grid case compared to the standard resolution case have increased in the presence of thermohaline forcing. The zonal velocity in the western boundary current is again too high, and the meridional velocity and transport is lower. When moving to the global case, the significance of these along- and cross-interface velocity differences will be examined for their impacts on global flow characteristics.

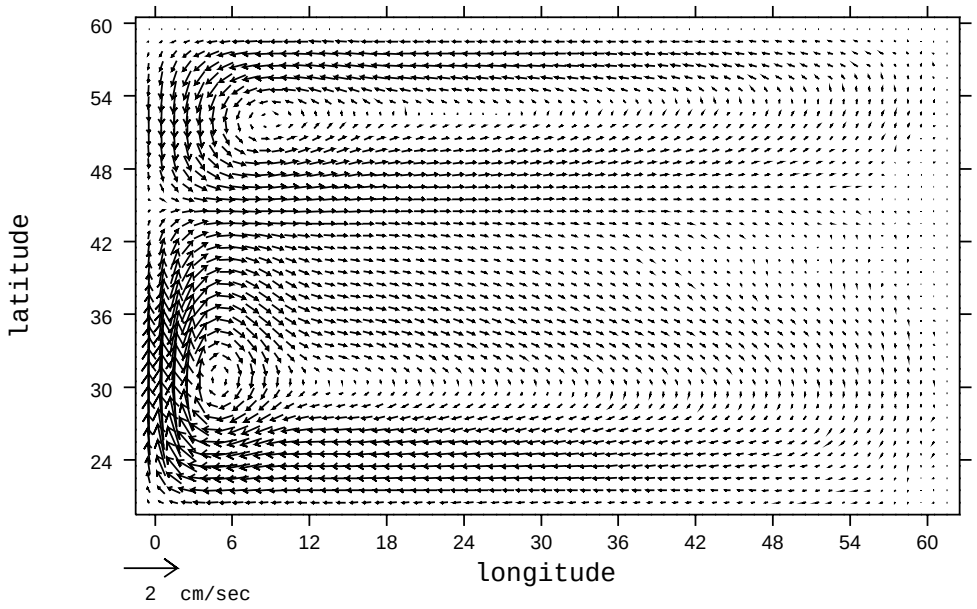

Figure 4.47: Barotropic velocity of the fully-forced, standard resolution basin.

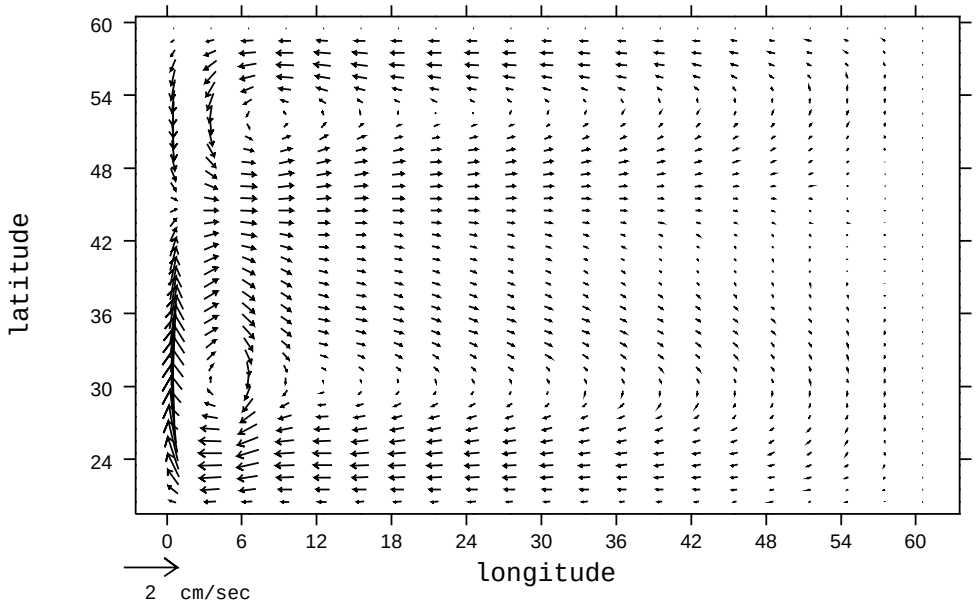

Figure 4.48: Barotropic velocity of the fully-forced, coarse resolution basin.

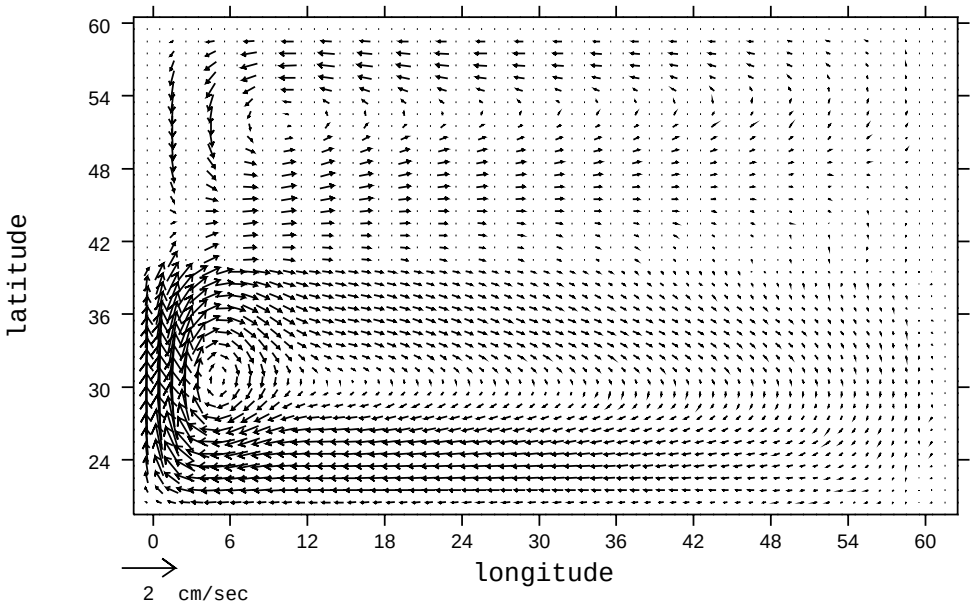

Figure 4.49: Barotropic velocity of the fully-forced, reduced grid basin.

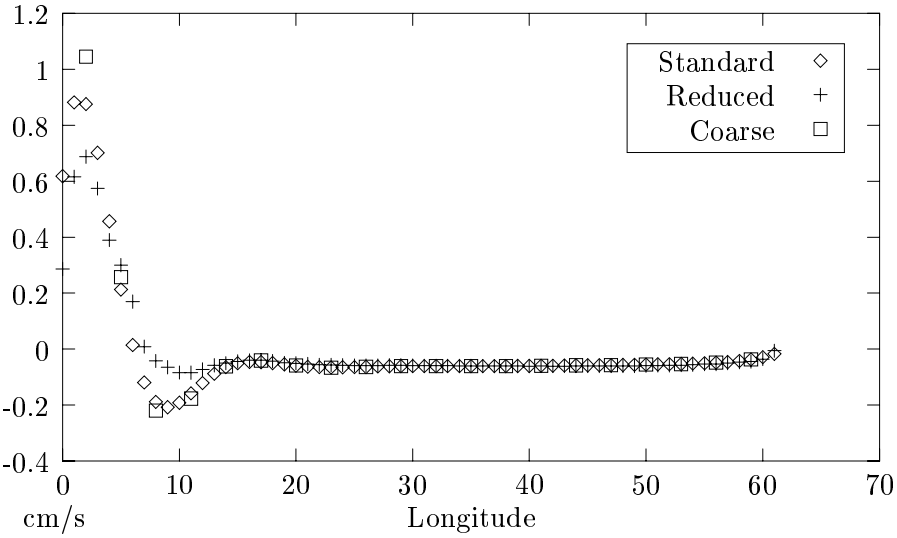

Figure 4.50: Meridional barotropic velocity of the fully-forced basin at the latitude row just south of the reduced grid interface location for various grid resolutions.

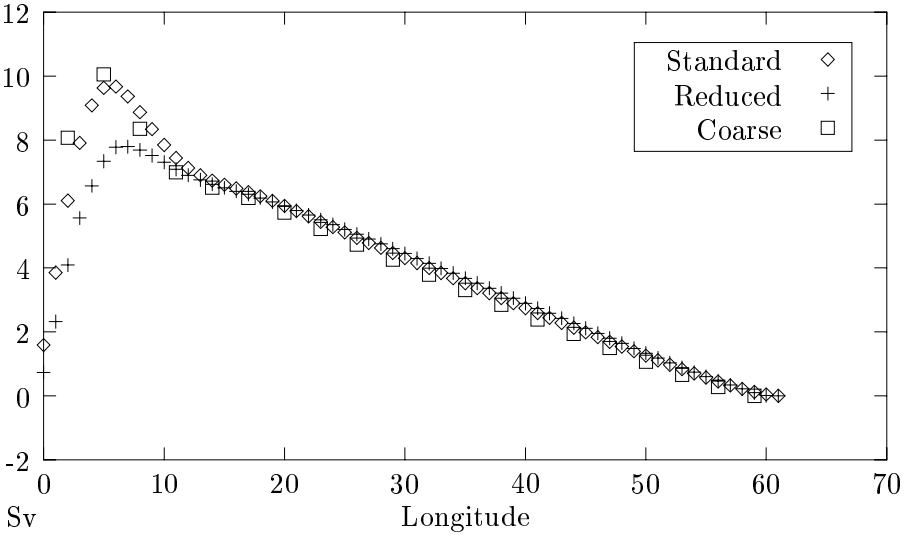

Figure 4.51: Integrated northward transport of the fully-forced basin at the latitude row just south of the reduced grid interface location for various grid resolutions.

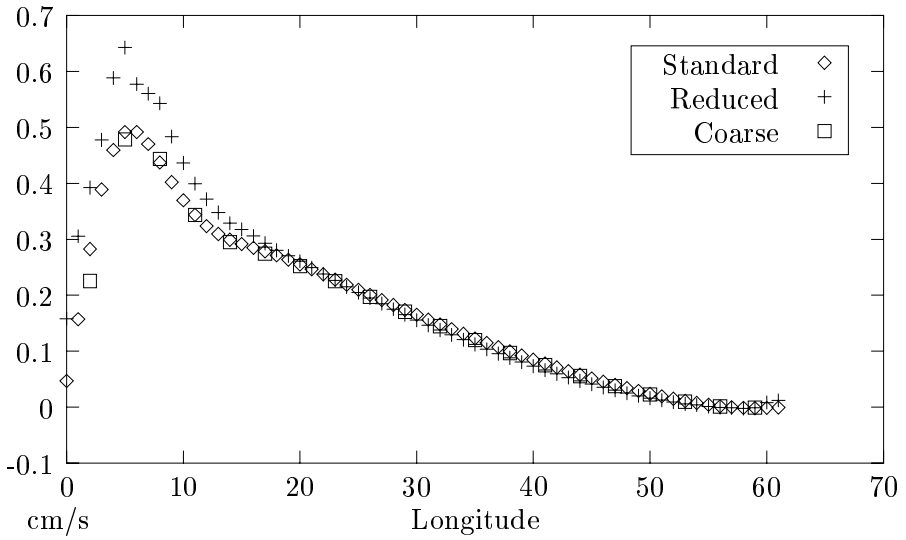

Figure 4.52: Zonal barotropic velocity of the fully-forced basin at the latitude row just south of the reduced grid interface location for various grid resolutions.

4.3.2 Surface Height

The surface height for the standard resolution case is shown in Figure 4.53. The deviations of the height from the mean are much larger than in the wind-driven only runs. The differences between the standard resolution surface height and that of the coarse and reduced grid cases are shown in Figure 4.54 and Figure 4.55. The coarse grid error is of the same character as the previous runs, the largest errors being at the western edge. The reduced grid error is smaller in comparison to the coarse error in this case and is more smooth. The zonal r.m.s. difference shown in Figure 4.56 also shows a smoother transition from the fine to the coarse regions of the reduced grid.

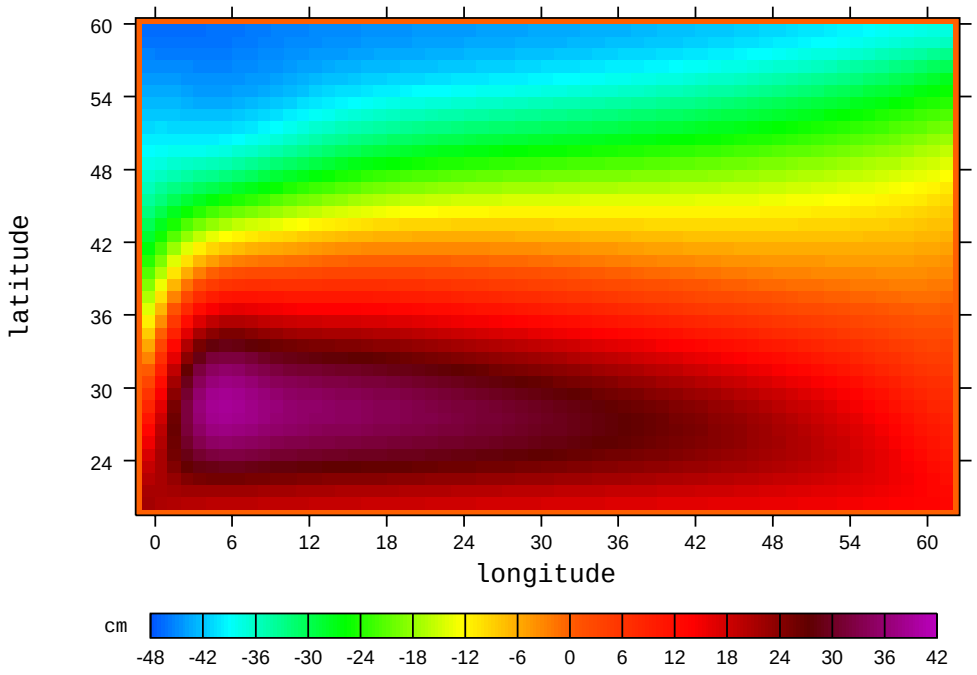

Figure 4.53: Surface height of the fully-forced, standard resolution basin.

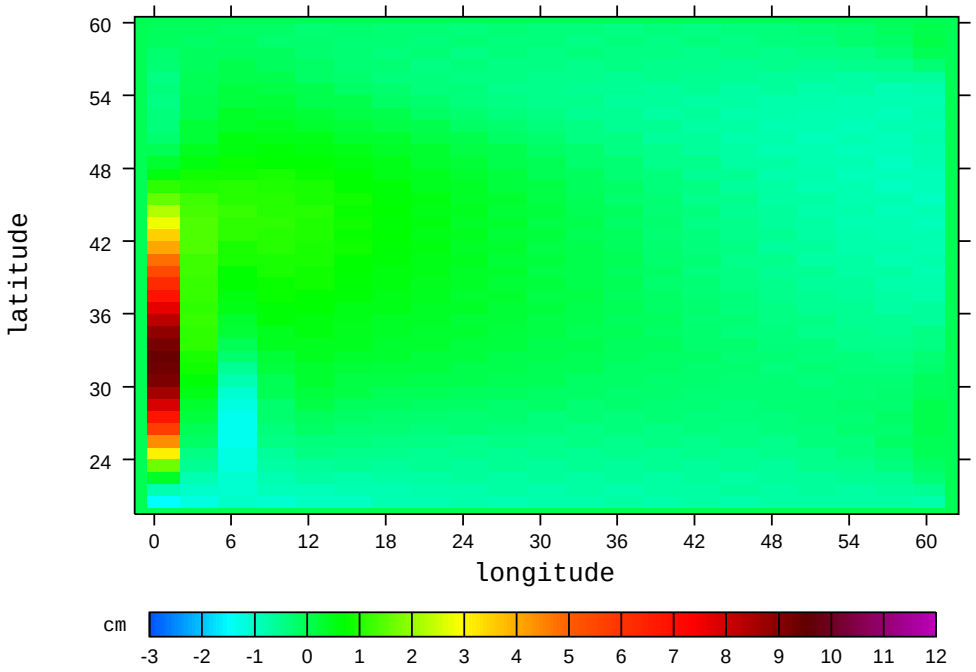

Figure 4.54: Difference between the surface height of the coarse resolution and the standard resolution cases of the fully-forced basin.

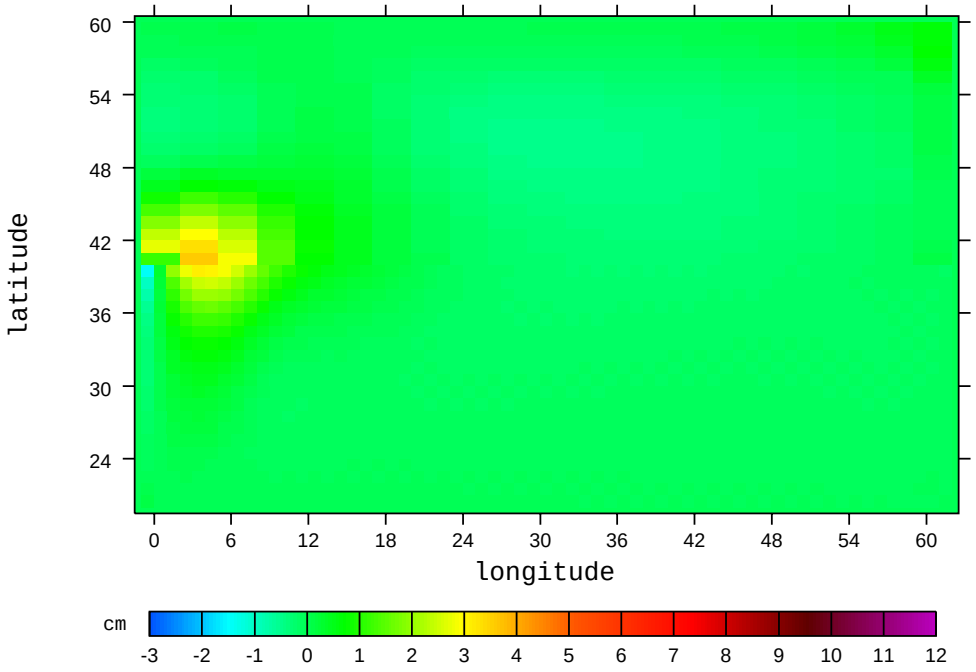

Figure 4.55: Difference between the surface height of the reduced grid and the standard resolution cases of the fully-forced basin.

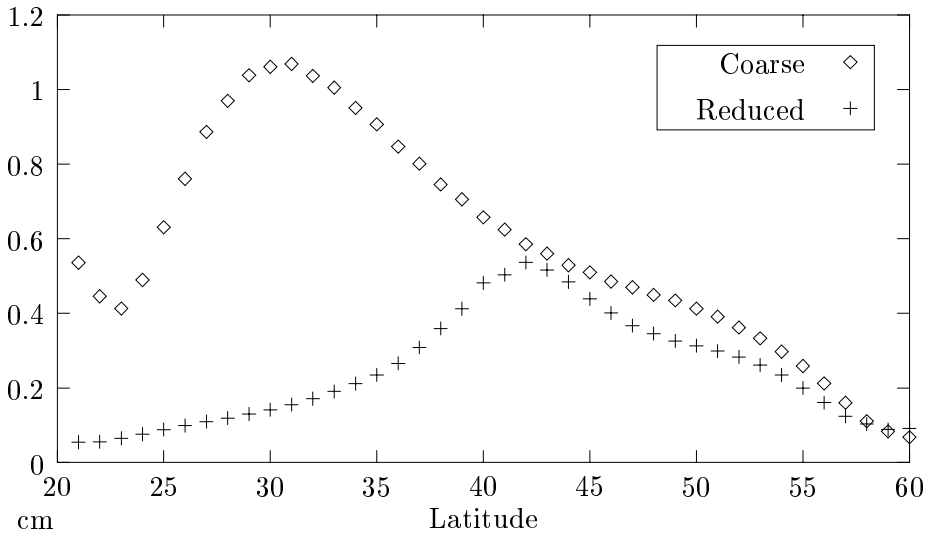

Figure 4.56: Zonal r.m.s. difference in surface height between various grid resolutions and the standard resolution case for the fully-forced basin.

4.3.3 Meridional Overturning

Figure 4.57 shows the meridional overturning for the standard resolution case. A much more complex circulation is revealed for this run than for the wind-driven only runs. Instead of simple overturning cells of opposite sign north and south, there is a more basinwide positive overturning that varies in strength considerably over the domain.

Differences between the coarse and reduced grid overturning and the standard case are shown in Figure 4.58 and Figure 4.59. The differences are also more complex, not corresponding so easily to a strengthening or weakening of the main overturning cells, but rather to a combination of this and shifts in the locations of maxima and minima. The coarse overturning has differences throughout the basin, as large as 20 percent in some locations. The reduced grid errors are generally under 10 percent and are restricted to the northern half of the basin that has the coarser resolution. Right at the interface location, the difference plot shows a small but significant difference of opposite sign than the coarse case. This is likely the effect of the interface velocity errors showing in the zonally averaged overturning.

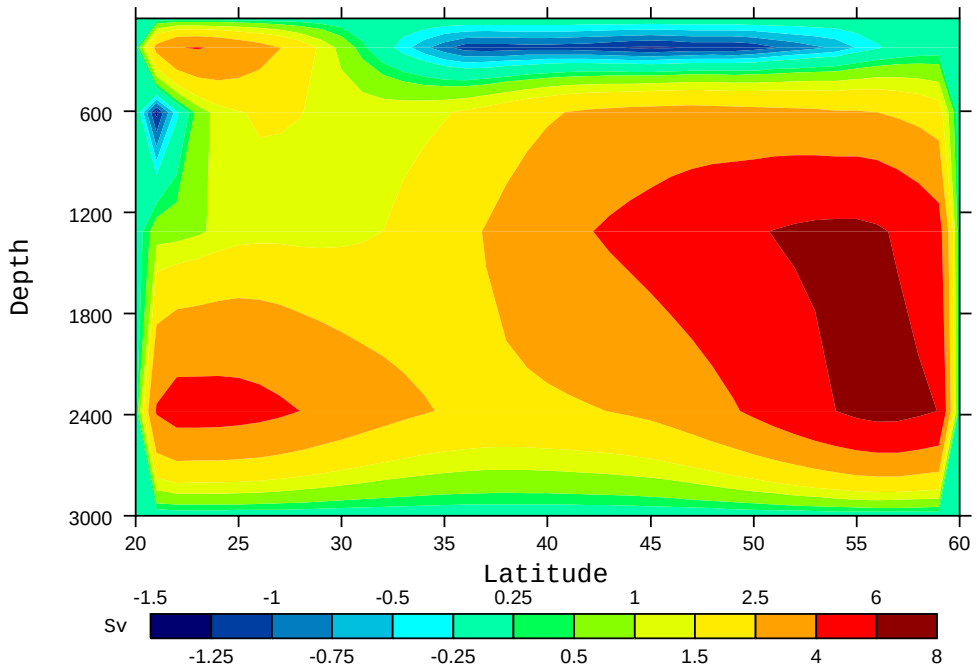

Figure 4.57: Meridional overturning of the fully-forced, standard resolution basin. Positive values indicate clockwise circulation.

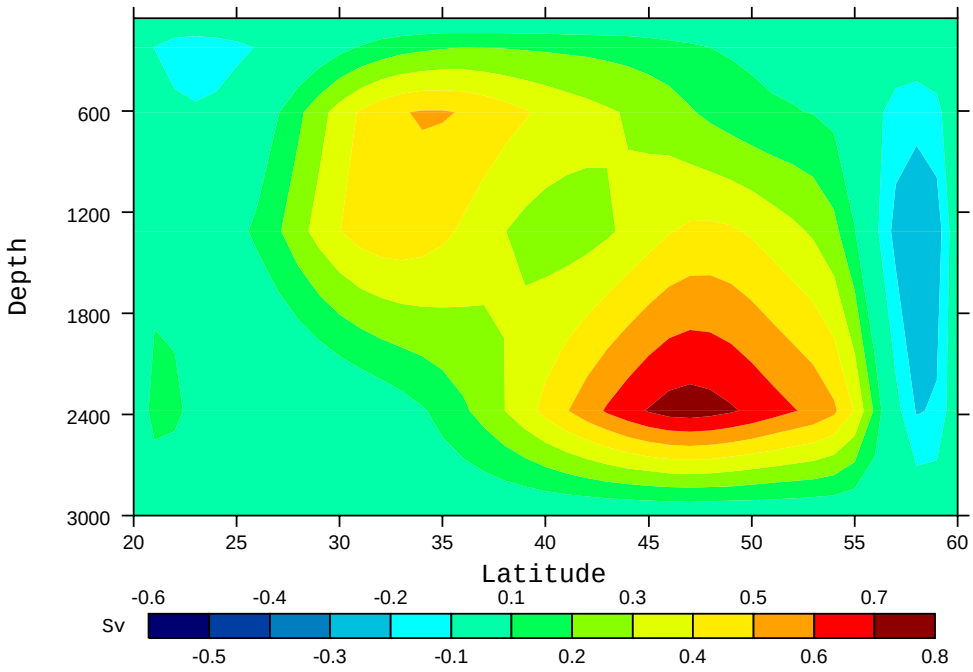

Figure 4.58: Difference between the meridional overturning of the coarse resolution and standard resolution cases of the fully-forced basin.

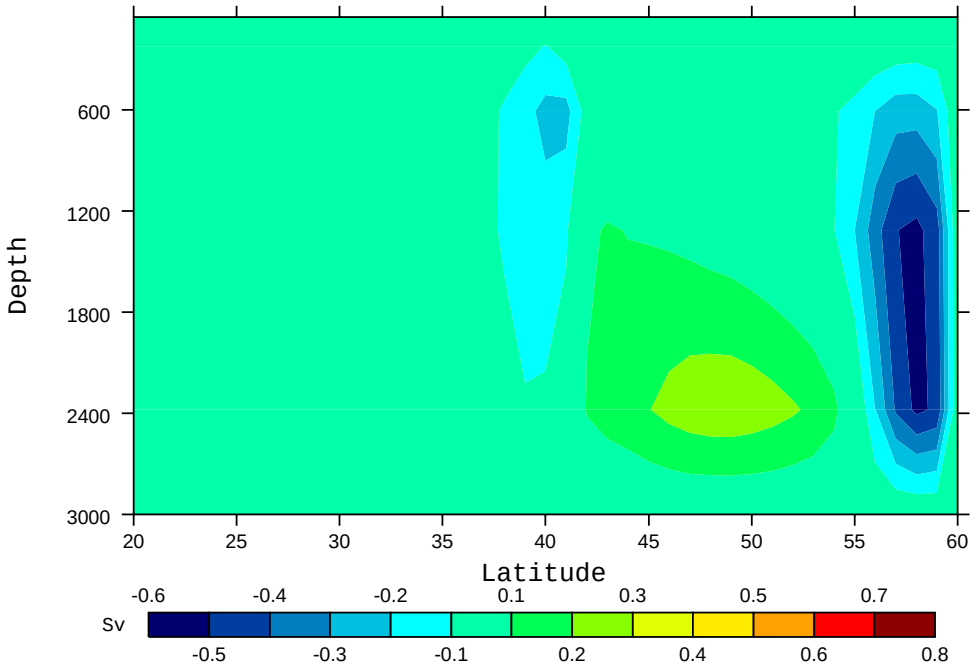

Figure 4.59: Same as Figure 4.58 except for the reduced grid and standard cases.

4.3.4 Temperature and Salinity

Figure 4.60 shows the temperature for the second depth level (223.57 m) of the standard resolution case. The initial condition of zonally constant temperature has evolved to resemble the two-dimensional spatial variations of the surface height, with a maximum in the southwestern region and a western boundary. Again, the standard case is used for comparison with the coarse and reduced grid cases. The resulting differences are shown in Figure 4.61 and Figure 4.62. Figure 4.63 through Figure 4.65 show the same for the model salinity. The differences are very similar in both spatial distribution and relative size to those of the surface height shown above.

Another view of tracer differences is shown in Figure 4.66 through Figure 4.69. Zonal r.m.s. differences of the coarse and reduced grid cases with respect to the standard case are shown for both the first depth level and the bottom depth level. At the surface, the reduced grid differences are very similar to those of the surface height. That is, they are much lower than those of the coarse grid throughout the southern part of the domain, very similar in the northern part, and increase rapidly when approaching the interface from the south with a noisy signal just north of the interface. At depth, however, the reduced grid differences are lower throughout the model domain. This indicates that the resolution in the deeper layers is less of a factor in determining the error than is the overall basin circulation. Both temperature and salinity were initialized with spatially uniform distributions in the deepest layer. The interaction with the upper layers through the basinwide circulation creates variation in these quantities, but they remain much more uniform than those near the surface and require fewer grid cells to resolve. Then since the reduced grid case produces a better basinwide circulation than the coarse grid case, as

shown by Figure 4.58 and Figure 4.59, the error in the deep layers is lower in the reduced grid case.

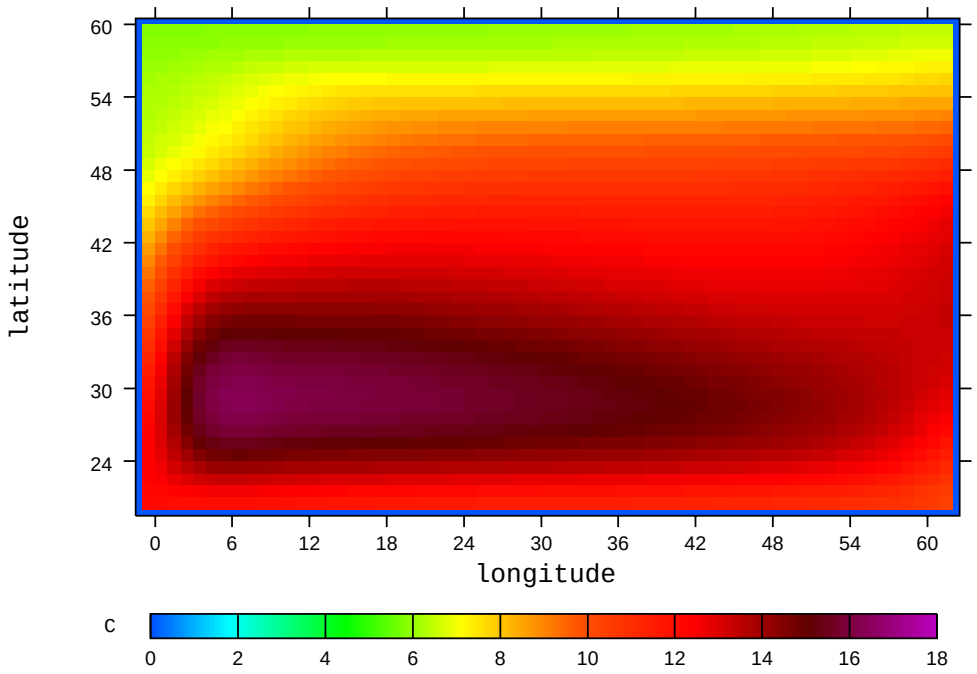

Figure 4.60: Temperature at the second depth level of the fully-forced, standard resolution basin run.

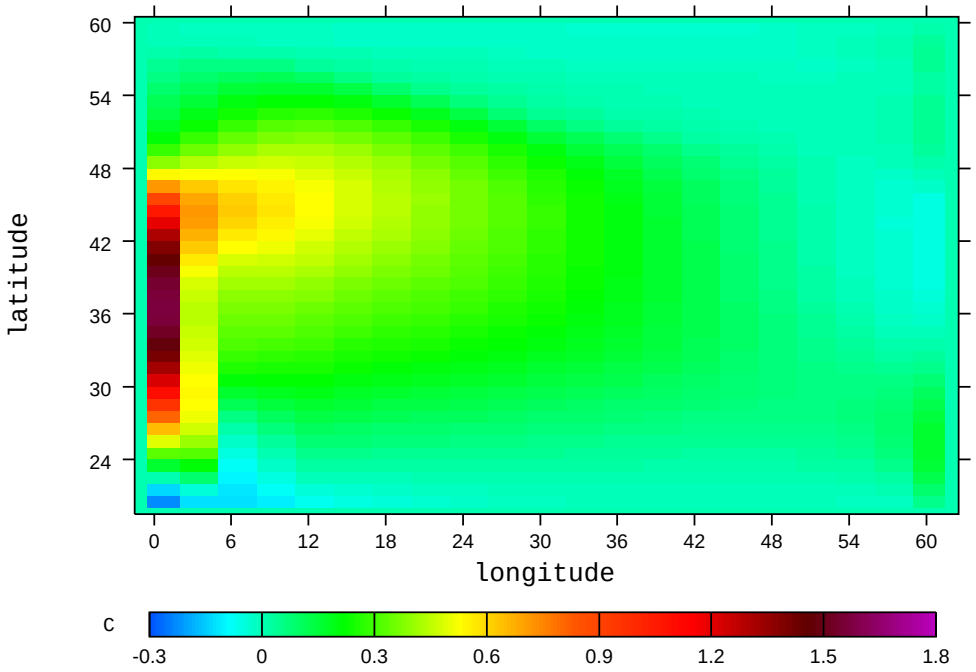

Figure 4.61: Difference between the temperature of the coarse resolution and standard resolution cases of the fully-forced basin at the second depth level.

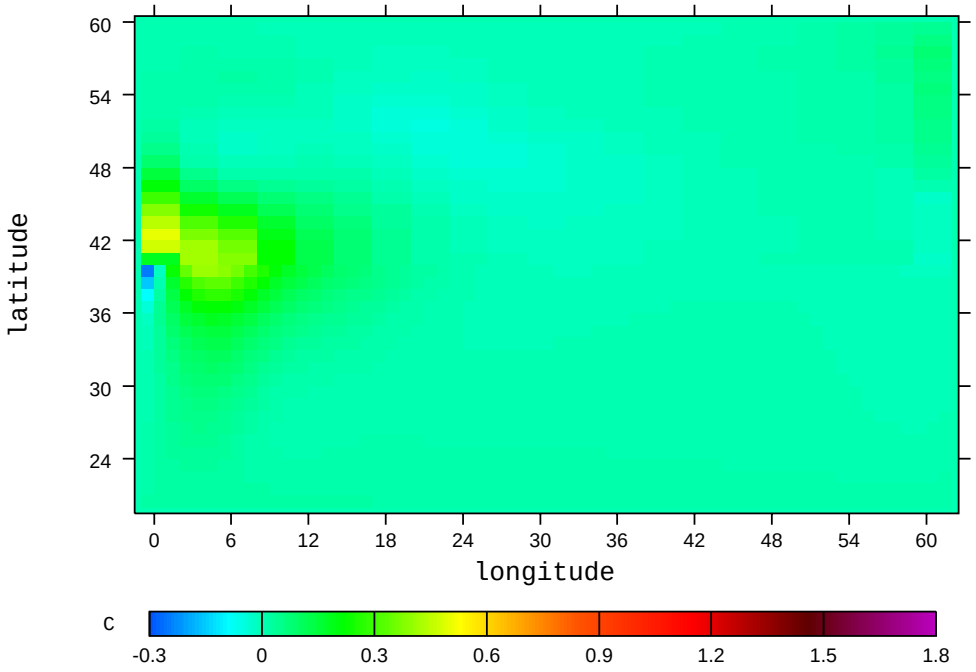

Figure 4.62: Same as Figure 4.61 except between the reduced grid and standard resolution cases.

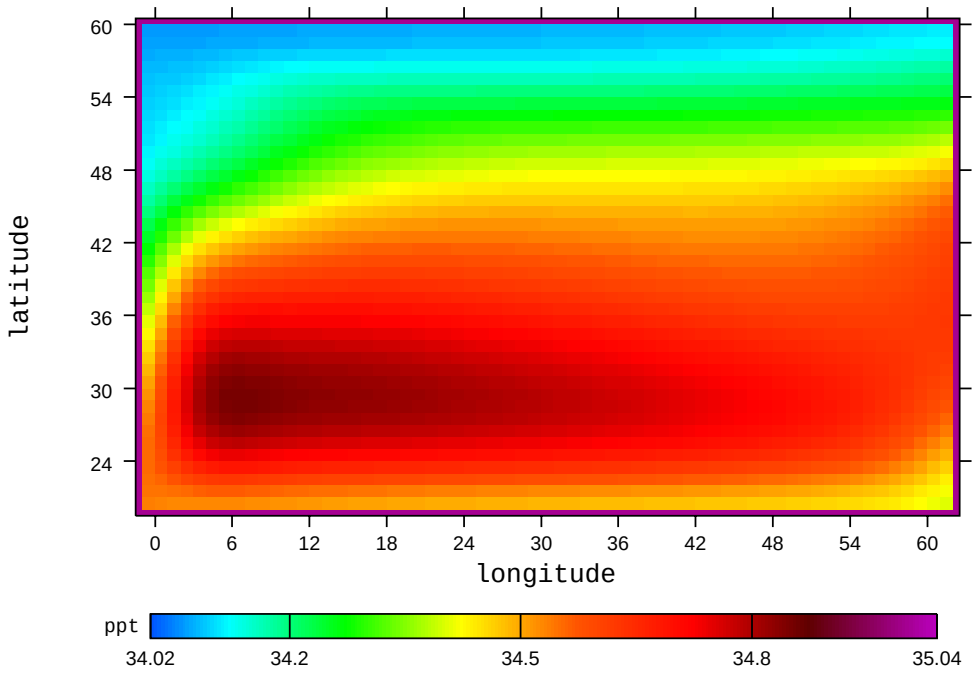

Figure 4.63: Salinity at the second depth level of the fully-forced, standard resolution basin run.

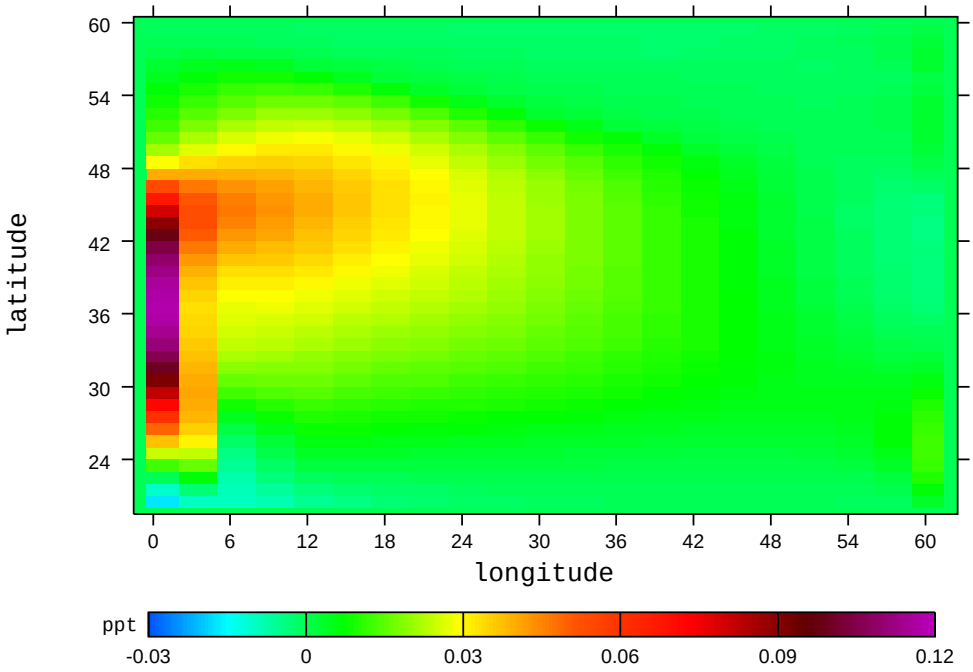

Figure 4.64: Difference between the salinity of the coarse resolution and standard resolution cases of the fully-forced basin at the second depth level.

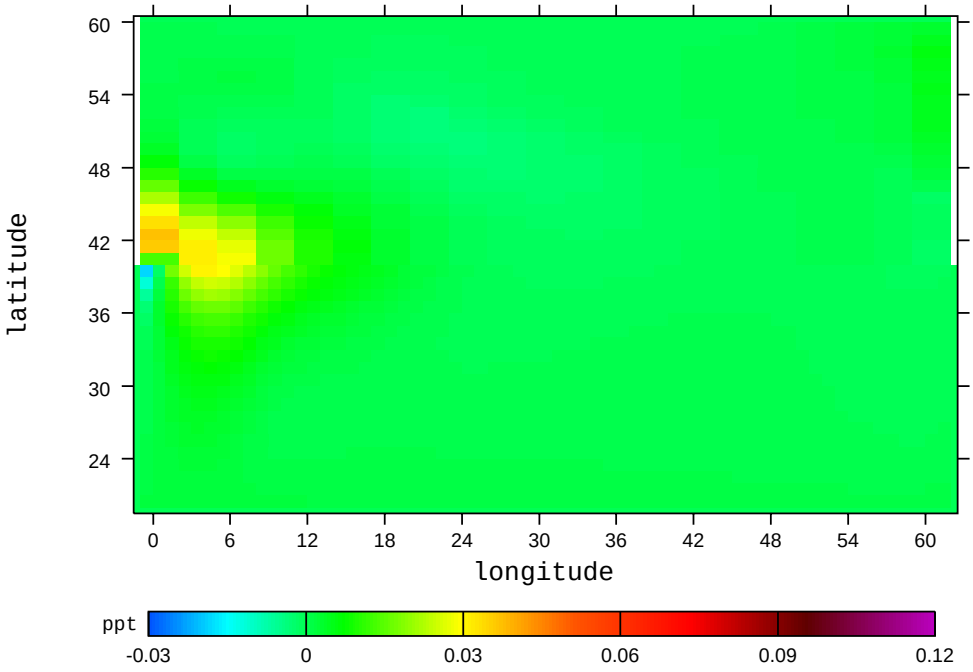

Figure 4.65: Same as Figure 4.64 except between the reduced grid and standard resolution cases.

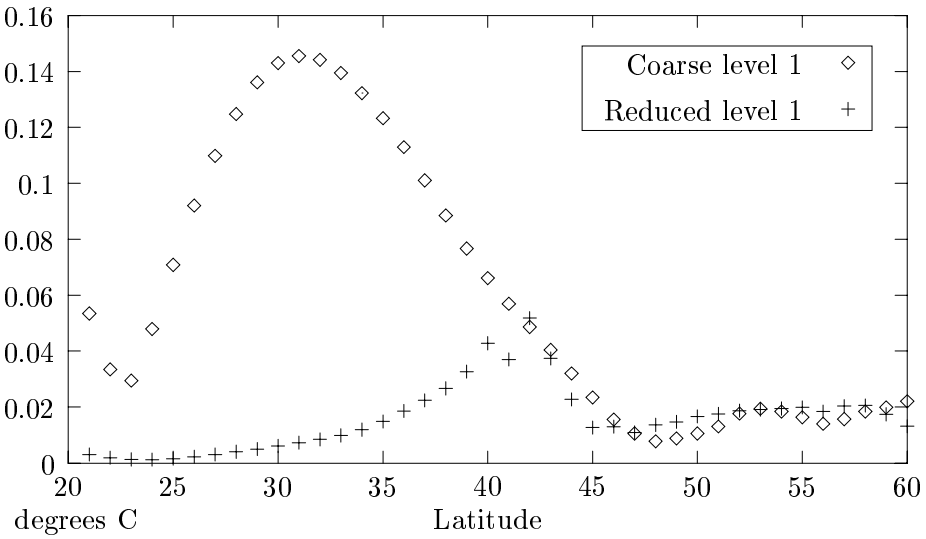

Figure 4.66: Zonal r.m.s. differences in temperature relative to the standard case for the coarse and reduced cases of the fully-forced basin at the surface depth level.

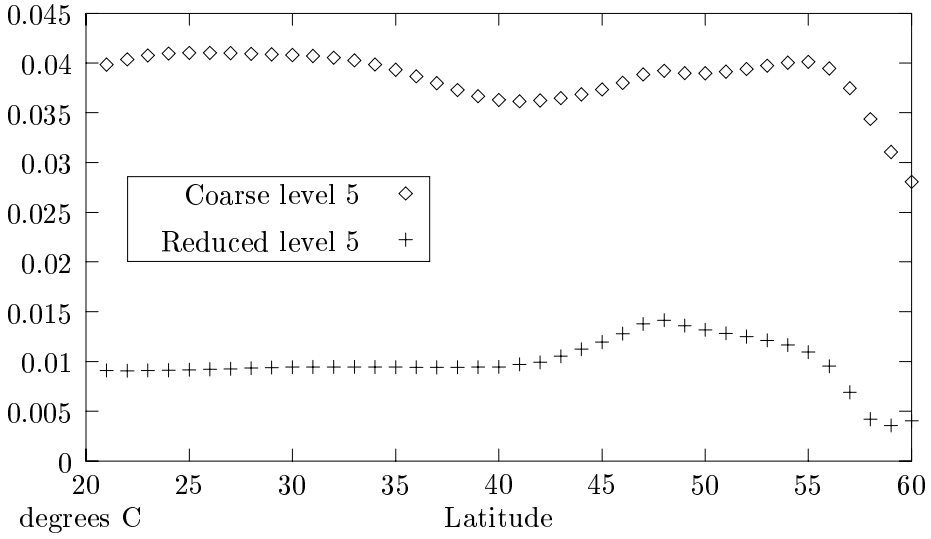

Figure 4.67: Same as Figure 4.66 except for the bottom depth level.

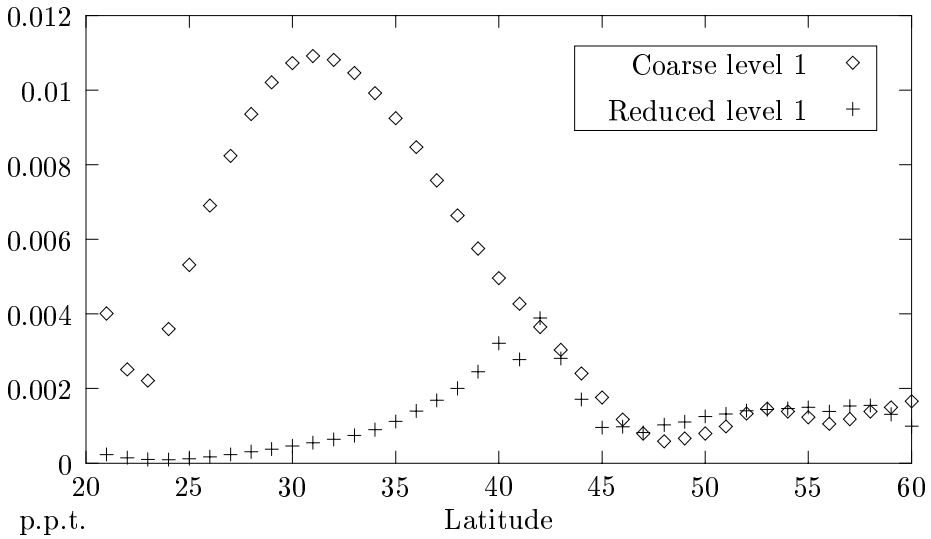

Figure 4.68: Zonal r.m.s. differences in salinity relative to the standard case for the coarse and reduced cases of the fully-forced basin at the surface depth level.

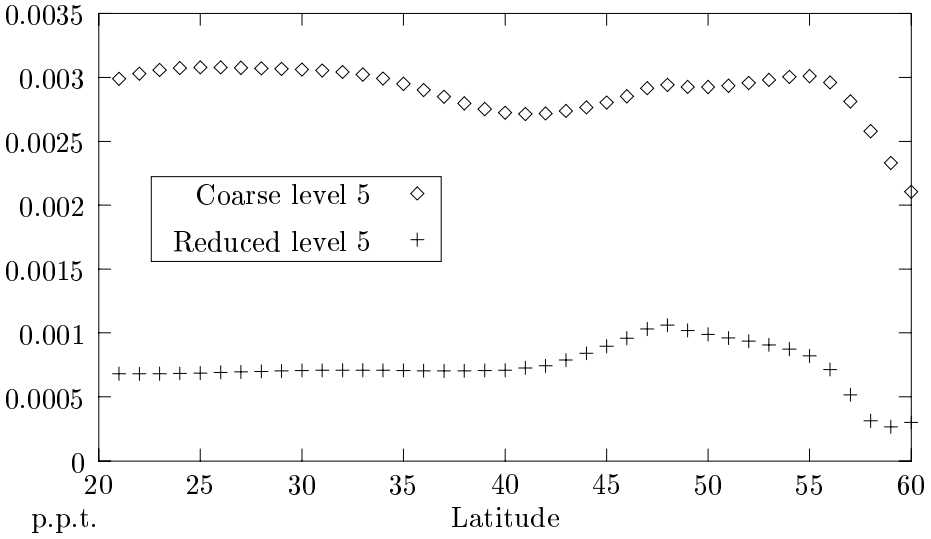

Figure 4.69: Same as Figure 4.67 except for the bottom depth level.

4.4 Summary of Limited Basin Runs

It is again noted that the simple mid-latitude basin described in this chapter is not in any way a targeted problem for the reduced grid method. It has been, however, a useful test case in which the influence of the method on the model equations can be demonstrated more clearly than in a larger and more complicated domain. Tracer fields, surface height, and meridional overturning show good agreement on their respective subgrids, with little degradation at the interface. The deeper layers show small differences with little sensitivity to the local reduced grid resolution. It has been shown that significant errors in both transport across an interface and velocity along interfaces can occur. Thus, when moving to the global runs, the impact of these types of errors should be noted.

Other cases are used in the literature for nested grid modeling which test the propagation of anomalies between coarse and fine grids. See [63] for tests with a barotropic modon and a baroclinic vortex and [30] for a test with a Gaussian surface height perturbation. However, the anomalies are generally mesoscale features of no more than a few hundred kilometers and not in the regime of application of the reduced grid method. Different choices are necessary for a global method with long timescales, notably that of conservation in preference to smoothness at interfaces.

It has been necessary to reduce the number of model options and grid resolutions tested as the complexity of the model configuration has increased the amount of computer time necessary for solution. Two types of reduced grid interface were included in the first tests. One of these was been dropped as a result of these practical considerations. Likewise, the first test began with a large number of interpolation options at the interface. As with the interface type, empirical results were used to select which options would be

used in the more resource consuming cases. As mentioned previously, the type-1 interface with linear interpolation will be the method used for the global runs. However, more work could be done to test options in the last case in this chapter, as well as the global runs of the next chapter.

Chapter 5

Global Simulations

The previous model runs were presented to get an idea of the behavior of the reduced grid method prior to runs with more complicated geometry and boundary conditions. However, the reduced grid method is particularly targeted at global domains where it can keep cells of more uniform size. It is in this context that gains in allowable timestep are expected while minimizing the influence of the grid interfaces.

This chapter will show results of the reduced grid model compared to the original model. A standard configuration of the models is detailed which is similar to published results of similar models. Comparisons between the standard and reduced grid model fields show the effects of the reduced grid on both overall results and details at the interfaces. Execution times are presented for a range of processor numbers and broken down by model component to discern the differences in efficiency of the codes on a parallel, distributed memory computer.

5.1 Model Configuration and Forcing

Two runs are compared in this chapter. The first uses a standard latitude-longitude grid with filtering, while the second uses the reduced grid described in Section 3.1.2. With the further exception of differing time steps, the simulations are otherwise identical. The specific definition of topography in the ocean model is covered in Section 2.3.2, and the modifications for the reduced grid model are given in Section 3.4.1. However, each run uses the same topography, shown in Figure 5.1. Most model parameters are directly comparable to those found in [16], which describes a set of model runs, one of which is directly comparable to the present runs.

The details of the definition of the reduced grid used were given in Section 3.1.2. Grid interfaces are located at 60 and 80 degrees north latitude, and at 60 degrees south latitude. The model domain only extends to 80 degrees in the south due to the presence of land at every point further south than that. Thus the standard grid model has a grid of 144 cells in longitude, 68 cells in latitude, and 23 depth levels, giving a 2.5×2.5 degree resolution. This gives 225,216 model cells for the standard grid. The reduced grid only requires 178,112 cells for the same resolution — a reduction of nearly 21 percent.

The topography is obtained by a multistep process. Data with a resolution of 1 degree is regridded to the 2.5 degree resolution of these runs by determining whether a cell contains more water or land. The regridded data is further modified to eliminate non-advective cells (those with a non-zero tracer point surrounded by four zero velocity points) and to ensure the separation of the Atlantic and Pacific basins in Central America. Following the procedure of [16], each ocean point is then given a minimum depth of 2500 m, and a smoothing function is applied multiple times. While the smoothing is not

Level	Depth to Tracer and U,V point	Depth to bottom of cell	Thickness
1	12.5	25	25
2	37.5	51.25	26.25
3	65	81.75	30.5
4	98.5	118.5	36.75
5	138.5	161.75	43.25
6	185	212.5	50.75
7	240	275	62.5
8	310	360	85
9	410	477.5	117.5
10	545	627.5	150
11	710	807.5	180
12	905	1017.5	210
13	1130	1262.5	245
14	1395	1557.5	295
15	1720	1922.5	365
16	2125	2350	427.5
17	2575	2800	450
18	3025	3250	450
19	3475	3700	450
20	3925	4150	450
21	4375	4600	450
22	4825	5050	450
23	5275	5500	450

Table 5.1: Vertical grid level definitions for the global run. All depths are in meters.

Parameter	value
Horiz. Viscosity	$10^9 \times (1 + 9 \times \cos\phi)$ cm^2/s
Horiz. Diffusivity	2×10^7 cm^2/s
Vert. Viscosity	20 cm^2/s
Vert. Diffusivity	0.2 (surface) – 1.3 (bottom) cm^2/s

Table 5.2: Mixing parameters for the global runs.

necessary from a numerical stability standpoint, as is often the case for a rigid-lid model, it is reasonable to limit topographical features to those that are well resolved by the grid. Then the depth data is discretized to the 23 level vertical grid defined in Table 5.1.

It is not strictly necessary to modify the resulting topography data for use with the reduced grid. The model will use the data specified at the model points each grid region defines and ignore the rest (see the reduction operation of Section 3.3). However, to eliminate the differences in topography from the comparison between the standard and reduced grid models, a further step is made for the topography data used on the standard grid. The depth information is averaged down to the level of the reduced grid, and then each set of standard grid cells is set to the level of the corresponding reduced cell. Using the terminology developed in Section 3.3, this corresponds to an average followed by an expansion. The resulting standard grid topography is coarser than required by the standard model grid. However, since the small scale model features will be filtered from the model variables at each timestep, it is not certain that the extra topographical information is significant. This approach of using identical topography in both the standard and reduced cases allows the isolation of changes in the solution due to the numerics of the reduced grid.

Table 5.2 gives the viscous and diffusive parameters used. The latitudinally dependent horizontal viscosity is employed to reduce the numerical errors caused by insufficient

vertical resolution in the equatorial region while not adversely affecting the stability requirements at high latitudes. The higher horizontal viscosity combined with the coarser model resolution compared with the limited basin test cases will likely yield smoother results and weaker boundary currents, possibly reducing the errors of the reduced grid method.

Surface forcing consists of specified wind stress and tracer restoring. The wind stress data is obtained from the monthly mean data of Hellerman and Rosenstein in [32], the annual mean of which is shown in Figure 5.2. Surface temperature and salinity are restored to data given by Levitus in [42] by the same process given in Section 4.3. Both of these data sets are regridded for the model by a process similar to and consistent with that of the topography data. Linear interpolation in time from these monthly mean data sets eliminates temporal discontinuities.

The model is initially at rest, i.e. all model velocities are zero. Initial temperatures and salinities are obtained from Levitus data. The zonal means of these initial conditions are shown by basin in Figure 5.3 and Figure 5.4. The definitions of the Atlantic and Indopacific basins, which will be used for many of the global model results, are shown in Figure 5.5. Since these initial conditions are provided from ocean observations, they are also used for comparing model results to the physical ocean. Reference will be made to the initial condition figures at times as the model results are given, though the emphasis will be on comparing the two model cases.

Model timesteps are different for tracers and velocities, as was discussed for the wind and thermohaline forced basin of Section 4.3. The basic tracer step is 6 hours, the baroclinic velocity step is 600 seconds, and the barotropic equations are subcycled 40 times per baroclinic step. In addition, the tracer step is larger in the deepest layers, increasing

to a value of 5 times the surface step at the bottom layer. This technique accelerates the convergence of the model solution to a steady state, since the deepest layers have small velocities and are the slowest to equilibrate. It is theoretically possible to converge to a different state with this method, but in practice this is not believed to occur. Again refer to [17] for details of the behavior of ocean models with this acceleration. Note that these timesteps, which are limited by numerical stability, are smaller than those in other published works with a similar model configuration, [16] in particular. The reason lies in the application of filtering in the LLNL ocean model.

Fourier filtering, as described in Section A.5, is applied for the standard, or base, case at latitudes of 60 degrees and higher. The tracers and baroclinic velocities are filtered at each timestep, with a reference latitude of 51 degrees for tracers and 54 degrees for the velocities. However, barotropic velocities and surface height are not filtered, thus limiting the barotropic timestep. It was not possible to increase the baroclinic step relative to the barotropic step by a larger number of subcycles. The instability that resulted from doing so seems to be due to a failure of the assumption that the forcing varies slowly enough during the barotropic subcycling to be held constant (see Section 2.2.3).

Other models filter the barotropic velocities and surface height in the same manner as the other model variables. While a larger allowable timestep will result, the tradeoffs in extra computation time can be large due to both the large number of barotropic timesteps and the difficulties in achieving load balance of the code on the parallel computational platform used. A quick implementation of filtering for the barotropic variables confirmed a stability limited step four times larger than the standard case. It also confirmed a significantly larger computational cost, enough to yield no net gain in performance. The implementation of the work of [46] to the filtering of these variables could be efficient

enough to be useful on parallel architectures, but this is speculative.

One goal of the reduced grid model was to allow larger timesteps while eliminating the need for filtering. This goal has been realized, as the reduced grid global case is stable with timesteps four times larger than the standard case, while entirely eliminating the need for filtering. Thus the tracer step is 1 day, the baroclinic step is 2400 seconds, and the barotropic equations still require only 40 subcycles per baroclinic step. The same acceleration of the tracer equations in the deep ocean is used.

Both the standard and reduced grid cases are integrated with the given timesteps for at least 4700 surface ocean years. By this time globally averaged temperature and salinity are changing less than 0.1 degree and 0.01 ppt per thousand years, respectively. At the end of this spinup, both cases are integrated with equal tracer and baroclinic steps and no deep ocean acceleration to minimize any time splitting errors.

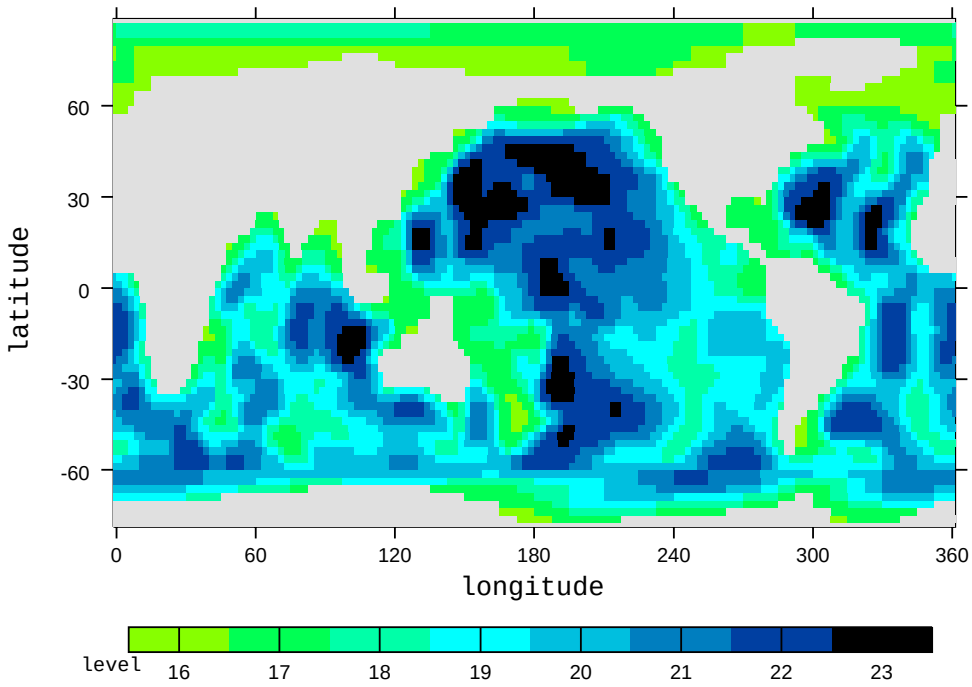

Figure 5.1: The topography used in the global runs. The value is the number of model levels at that location. See Table 5.1 for the corresponding depths of the levels.

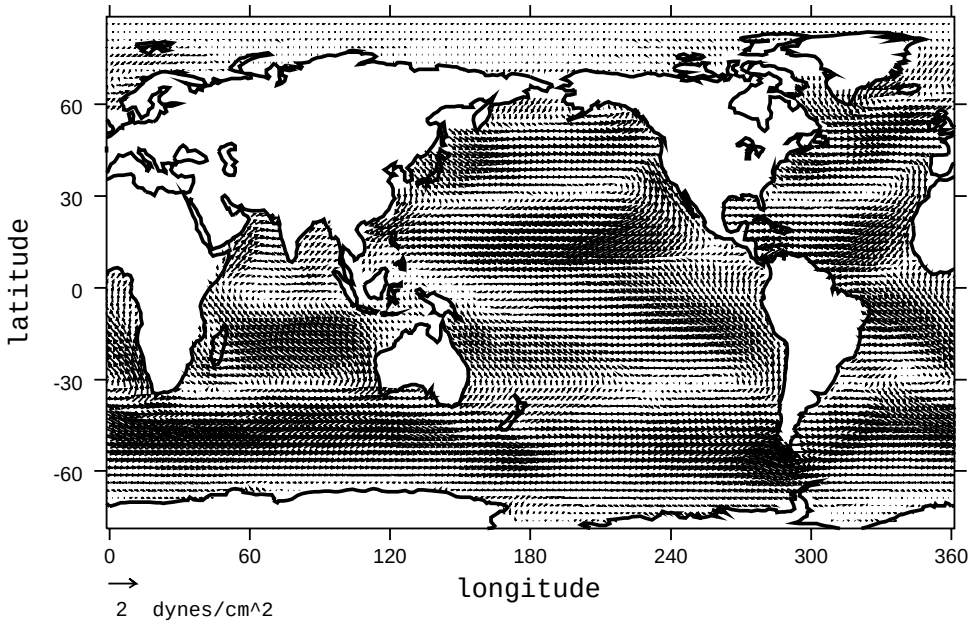

Figure 5.2: Annual mean surface wind stress data from Hellerman and Rosenstein

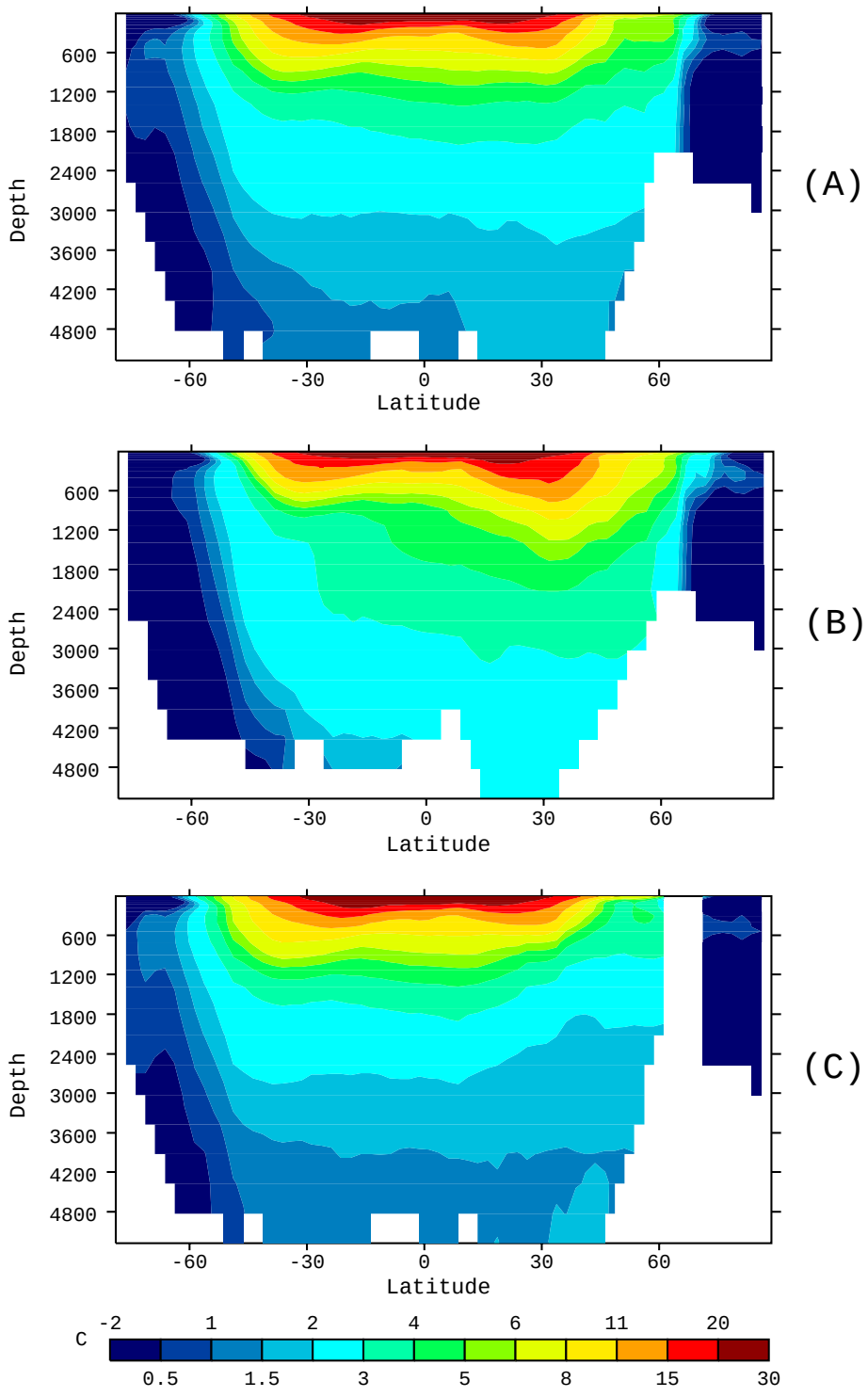

Figure 5.3: Zonal mean initial temperature from Levitus. (A) Global. (B) Atlantic. (C) Pacific.

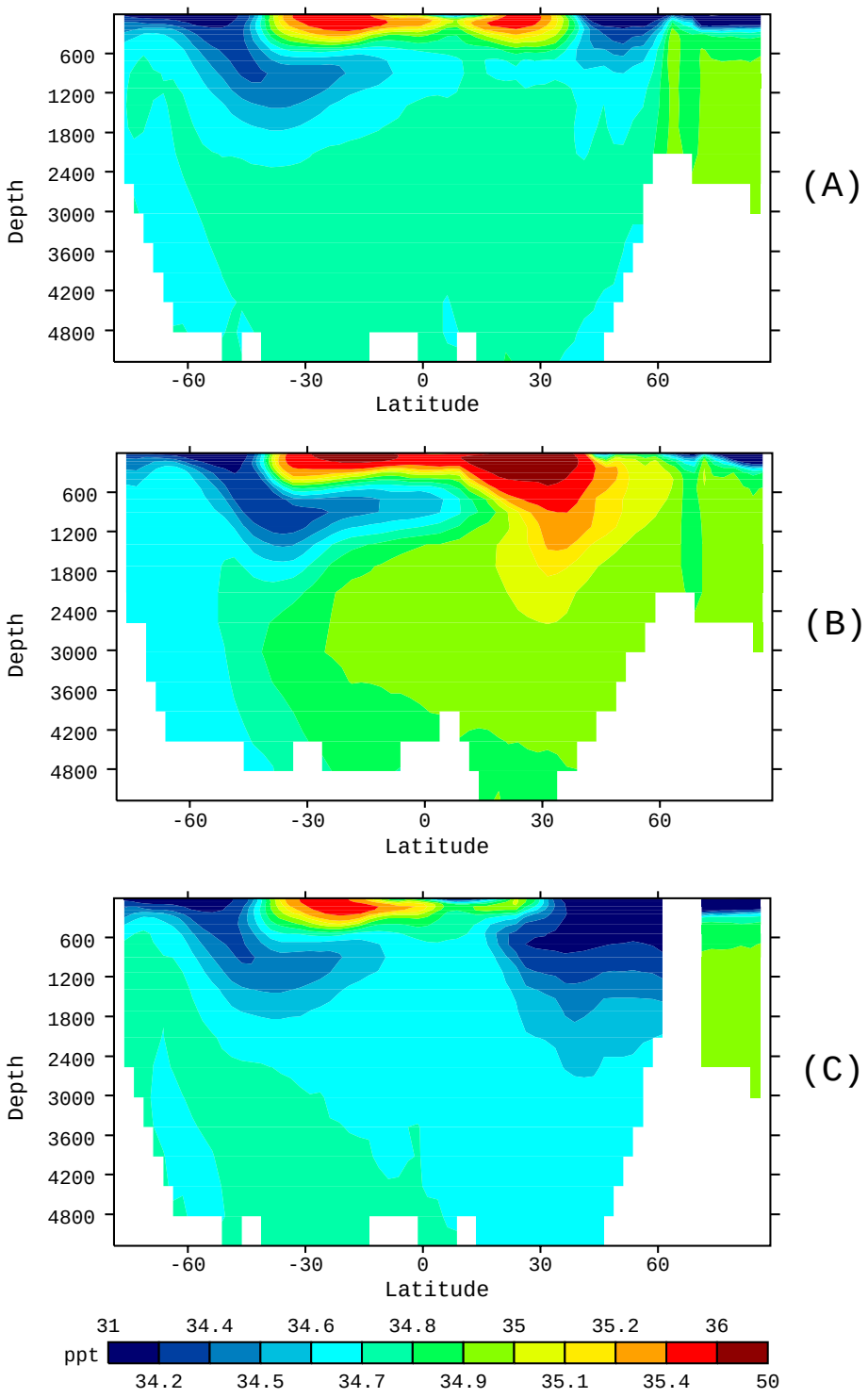

Figure 5.4: Zonal mean initial salinity from Levitus. (A) Global. (B) Atlantic. (C) Pacific.

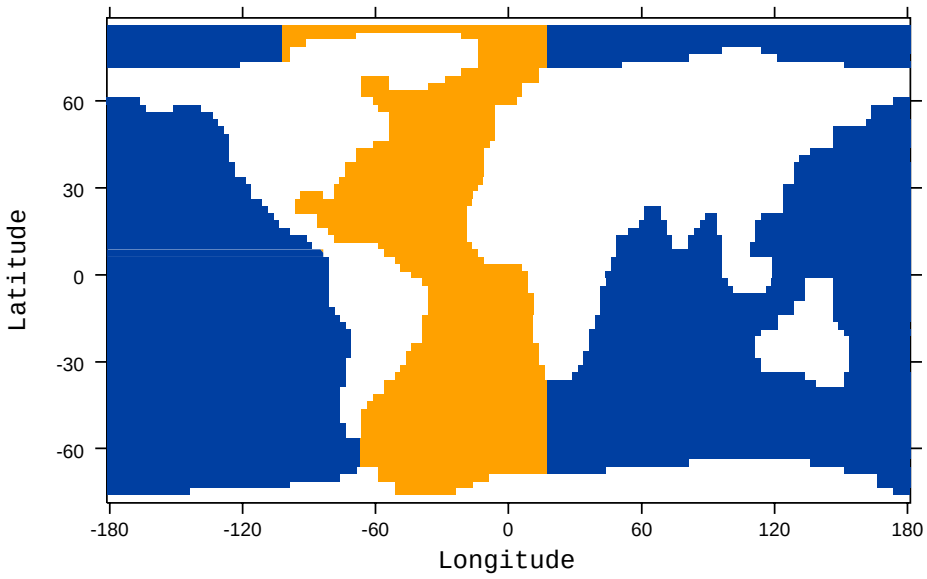

Figure 5.5: Definition of basins used for zonal mean calculations. Global domain is split into Atlantic and Indopacific.

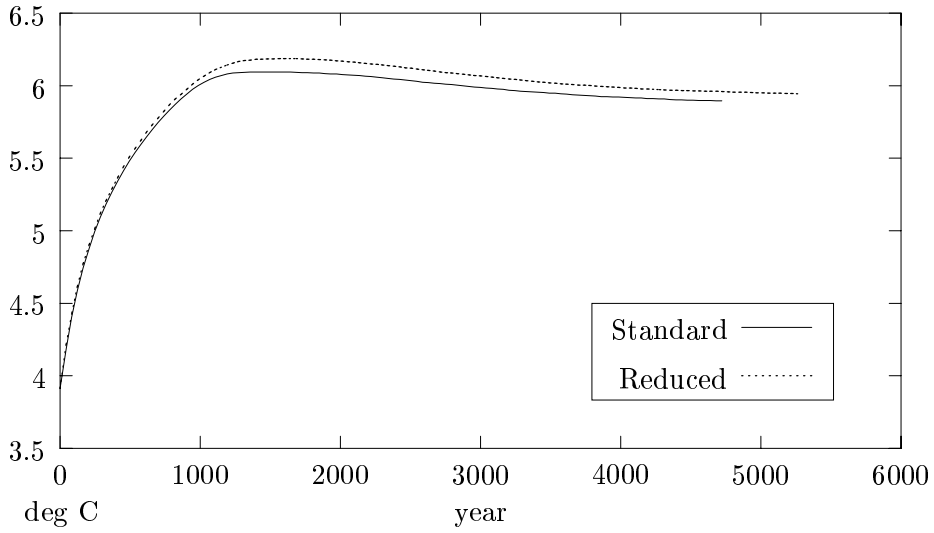

Figure 5.6: Global average temperature versus time.

5.2 Temperature and Salinity

The time history of the global mean temperatures and salinities are shown in Figure 5.6 and Figure 5.7, respectively. Both the temporal evolution and the final values are very similar between the two cases and to similar published cases. The slightly longer time for the reduced grid case to reach an equilibrium value is most likely due to the different timesteps used rather than to differences in the numerics. The final values of salinity are both roughly 34.5 ppt. Final temperatures show some difference, with the standard case having value of 5.89 degrees and the reduced case a value of 5.95 degrees. This difference is very small compared to deviation from the initial, data derived condition of around 4 degrees. The excessively warm and fresh ocean resulting from these model runs is typical of Bryan-Cox type ocean models with simple horizontal and vertical mixing schemes (see [12], [16], [24], and [21]).

More detailed results of temperature are shown in the zonal mean plots of Figure 5.8

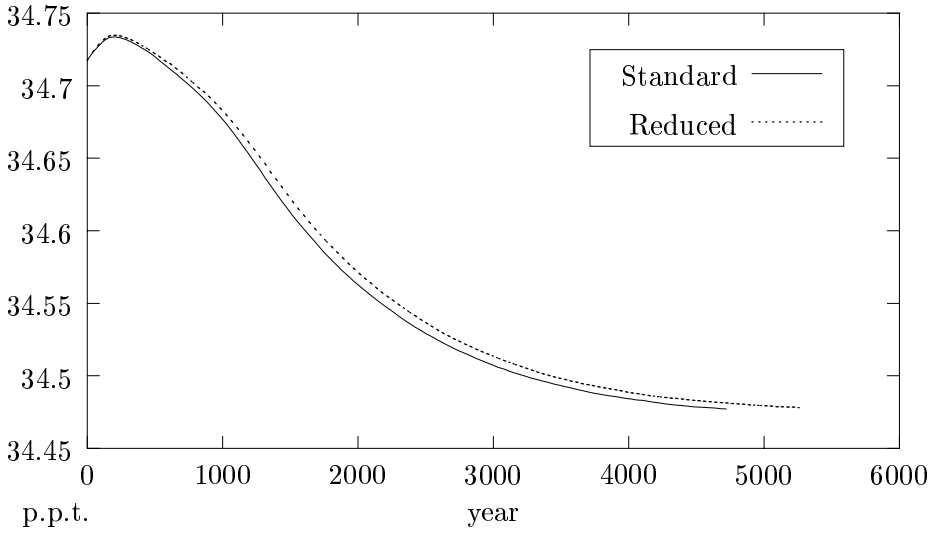

Figure 5.7: Global average salinity versus time.

through Figure 5.10. Comparison with the initial conditions shows that the deeper regions are much too warm and that the thermocline is much too diffused. The global zonal mean difference of temperature clearly shows that the reduced grid case is warmer than the standard case throughout most of the domain. The largest differences of over a tenth of a degree occur in the southern and deep ocean as well as near the bottom at the location of the interface at 60 degrees north. The Pacific basin mean shows the same difference in the southern and deep ocean, but the presence of mostly land at 60 degrees north eliminates the effect of the reduced grid interface. However, in the Atlantic the difference is positive nearly everywhere, and the effect of the interface below 2000 m is nearly 0.2 degrees. The topography at the 60 degree north reduced grid interface changes by one level across more than half of the ocean cells at that latitude. The differences in vertical velocity and convective adjustment due to the change in reduced grid resolution likely are made even more prominent by this topographical feature. The differences are, however, fairly small and don't affect the overall temperature distribution significantly.

Analogous plots for salinity are shown in Figure 5.11 through Figure 5.13. Comparison with the initial conditions shows that the deep ocean is too fresh as well as being too warm. Like the temperature, an increase in salinity from the standard to the reduced case is seen in the southern and deep ocean as well as at the 60 degree north interface. The salinity feature at 60 degrees north in both the standard and reduced global zonal means of Figure 5.11 is not due to the reduced grid interface. It is an artifact of the averaging process due to the presence of land in the Pacific basin there combined with the relatively large difference in salinity structure between the two basins. The difference of up to 0.025 parts per thousand is apparently caused by the interface in the Atlantic basin in the same manner as the temperature difference described above. Again, while the structure of the salinity differences is obviously caused by the reduced grid method, the magnitude of the difference is such that there is little effect on the overall distributions.

The surface tracers are restored to observed values as described above. However, if there are large differences between the standard and reduced grid surface tracer values, there is an implied difference in the transfer of heat and fresh water into and out of the ocean surface. Figure 5.14 and Figure 5.15 show the values and differences of the surface temperature and salinity. The largest differences occur in the regions of the reduced grid resolution, as expected. However, they are largely localized to individual cells, and those cells don't show systematic differences but both positive and negative differences instead. The cause of the largest differences being isolated in individual cells is most likely the convective adjustment algorithm, which operates only when density values in a single column reach unstable conditions (see Section 2.3.5). When convection occurs, comparatively high levels of mixing can occur. The large individual surface differences are located in regions where convection is expected to take place, e.g. near Antarctica and in

the North Atlantic. And since the columns of the reduced grid are 3 to 9 times larger than those of the standard grid, it is expected that convection will occur in a substantially different manner.

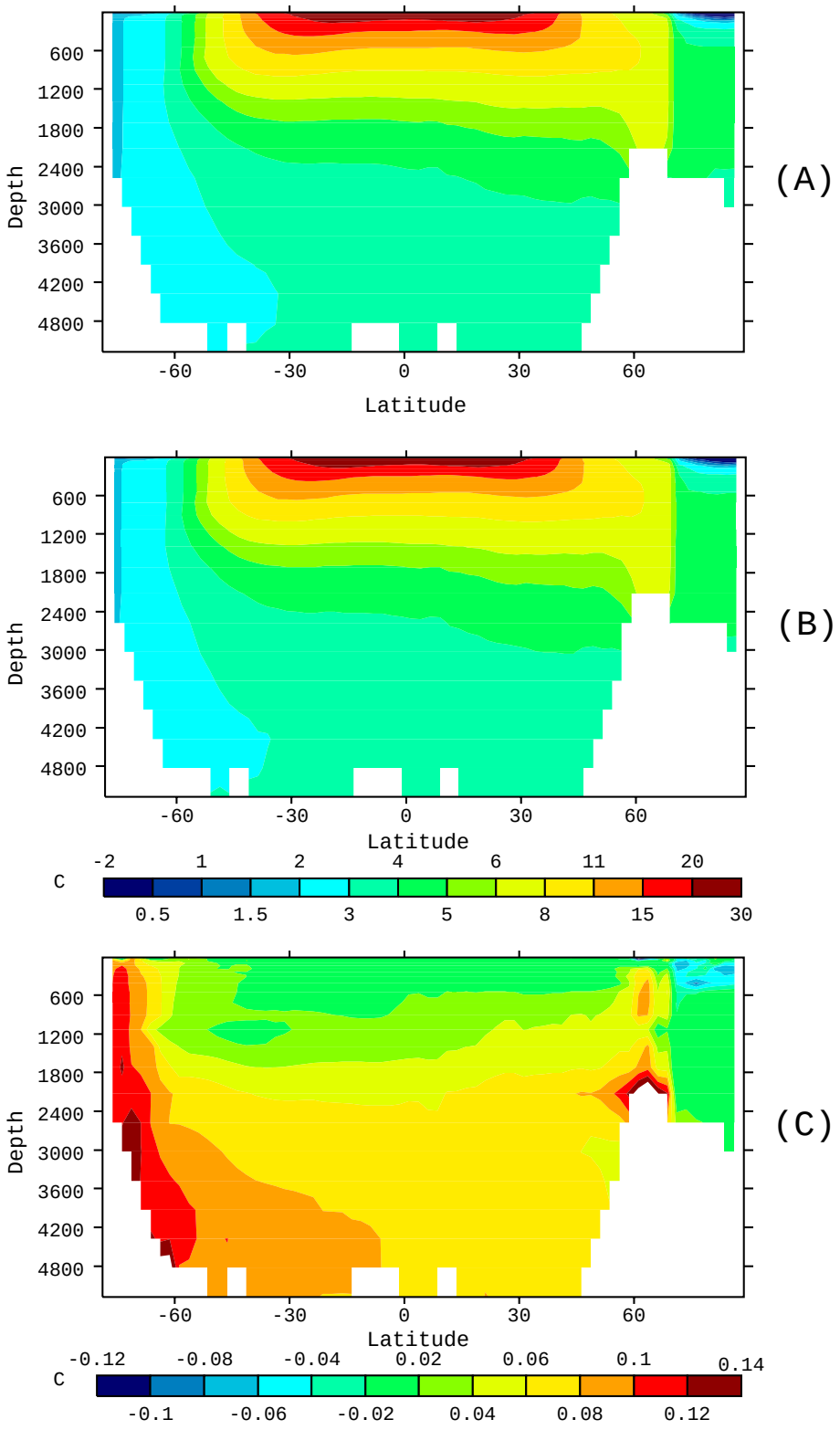

Figure 5.8: Global zonal mean temperature. (A) Standard grid. (B) Reduced grid. (C) Reduced - Standard grid.

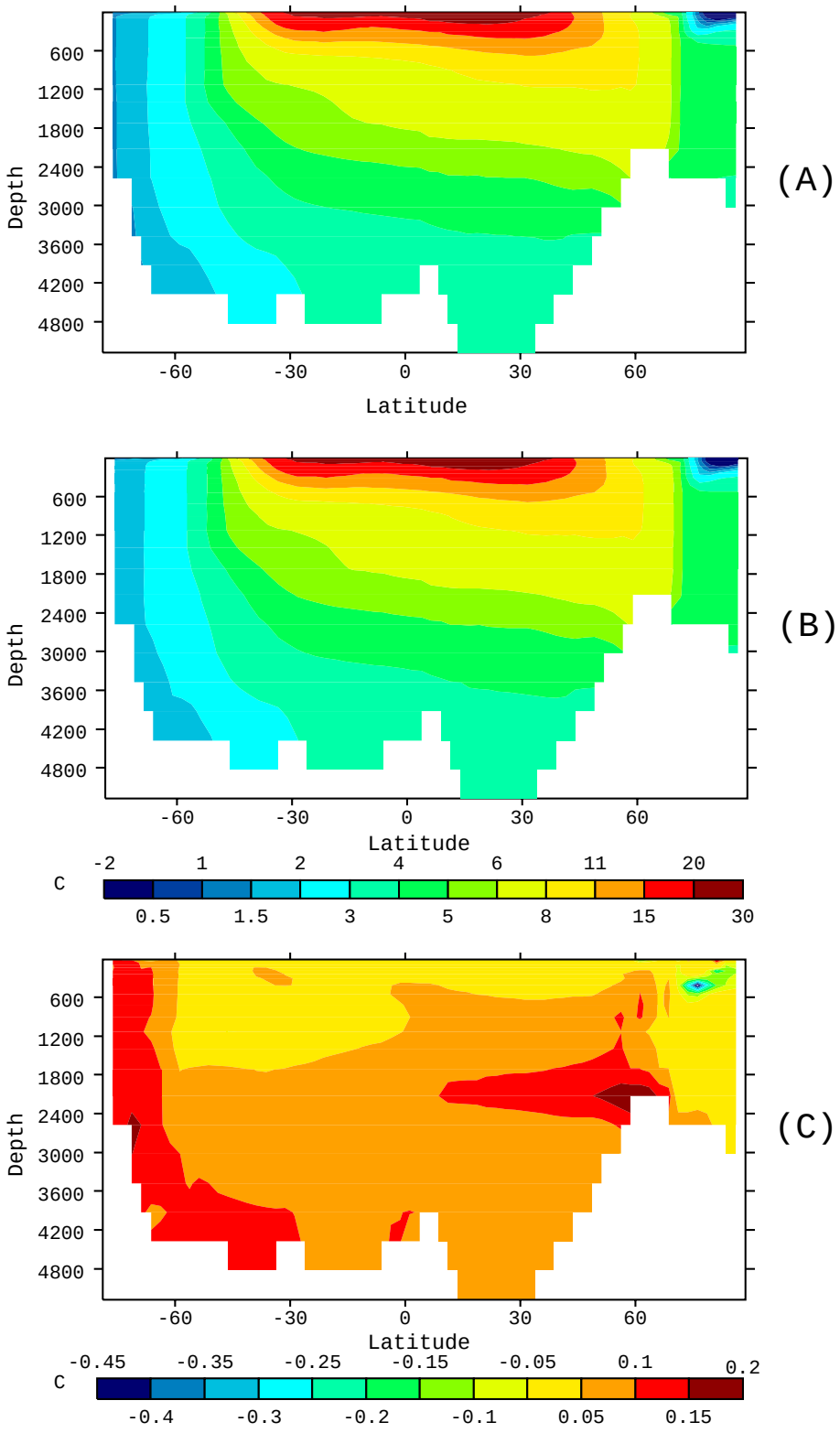

Figure 5.9: Atlantic zonal mean temperature. (A) Standard grid. (B) Reduced grid. (C) Reduced - Standard grid.

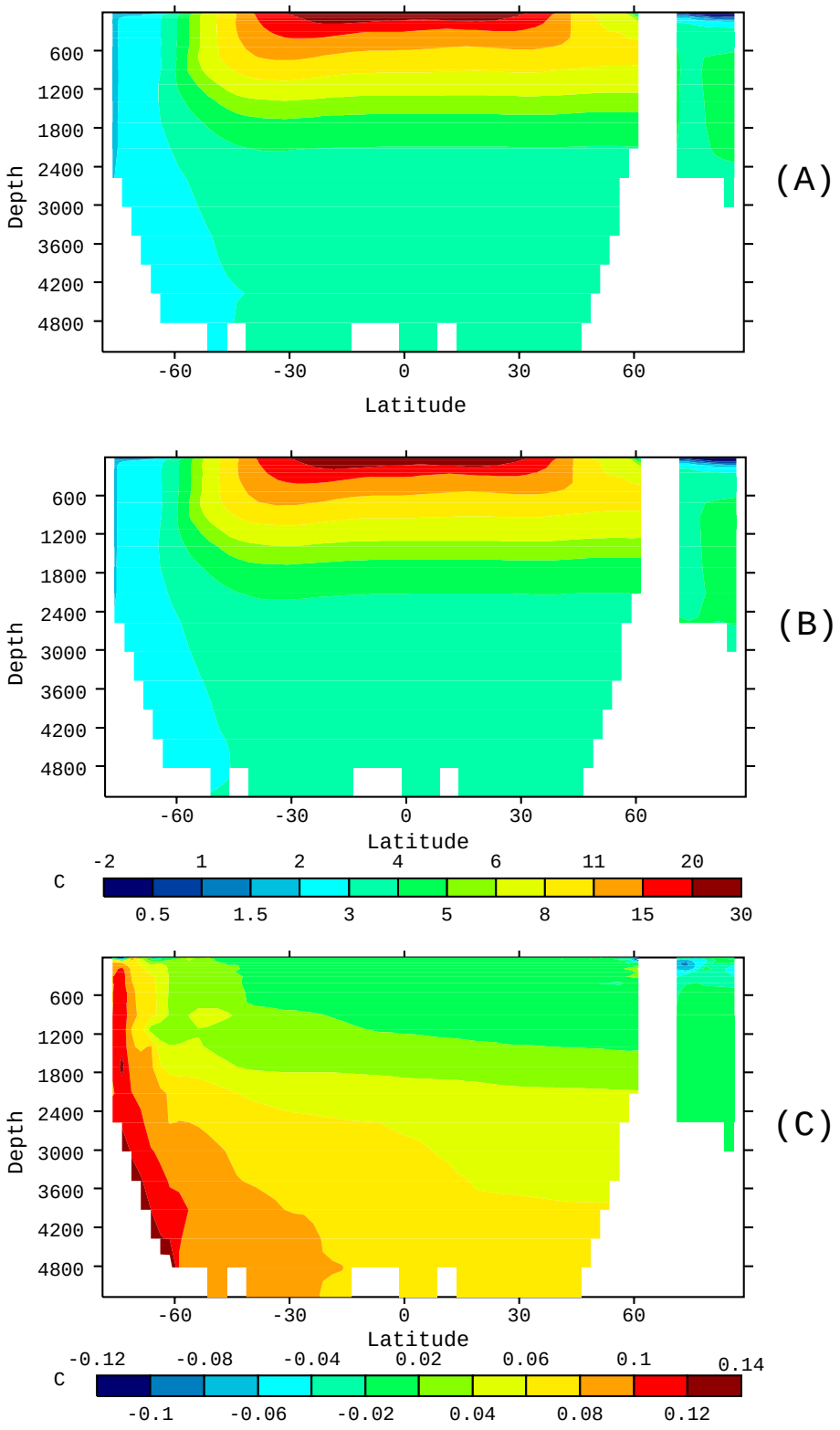

Figure 5.10: Pacific zonal mean temperature. (A) Standard grid. (B) Reduced grid. (C) Reduced - Standard grid.

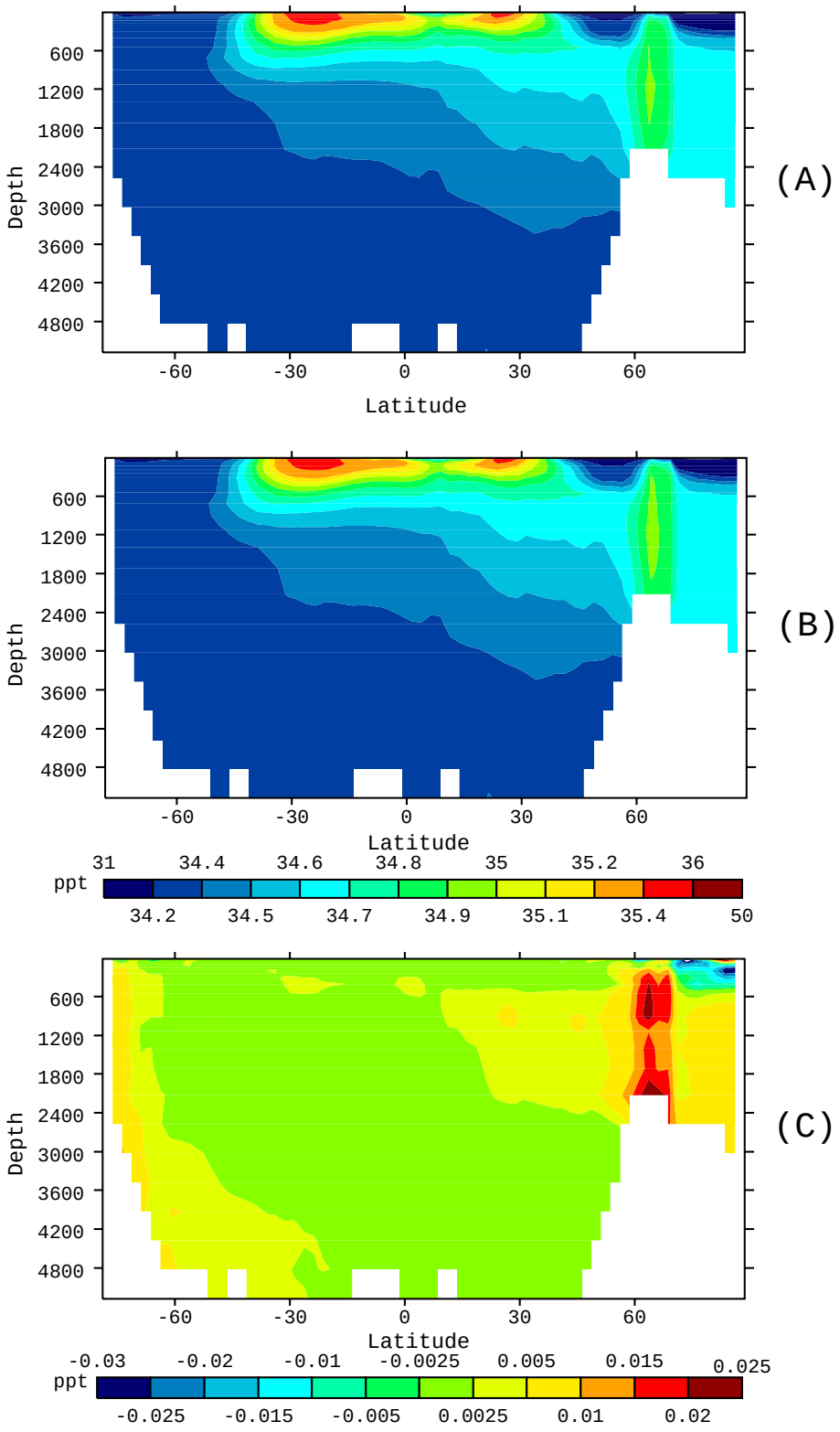

Figure 5.11: Global zonal mean salinity. (A) Standard grid. (B) Reduced grid. (C) Reduced - Standard grid.

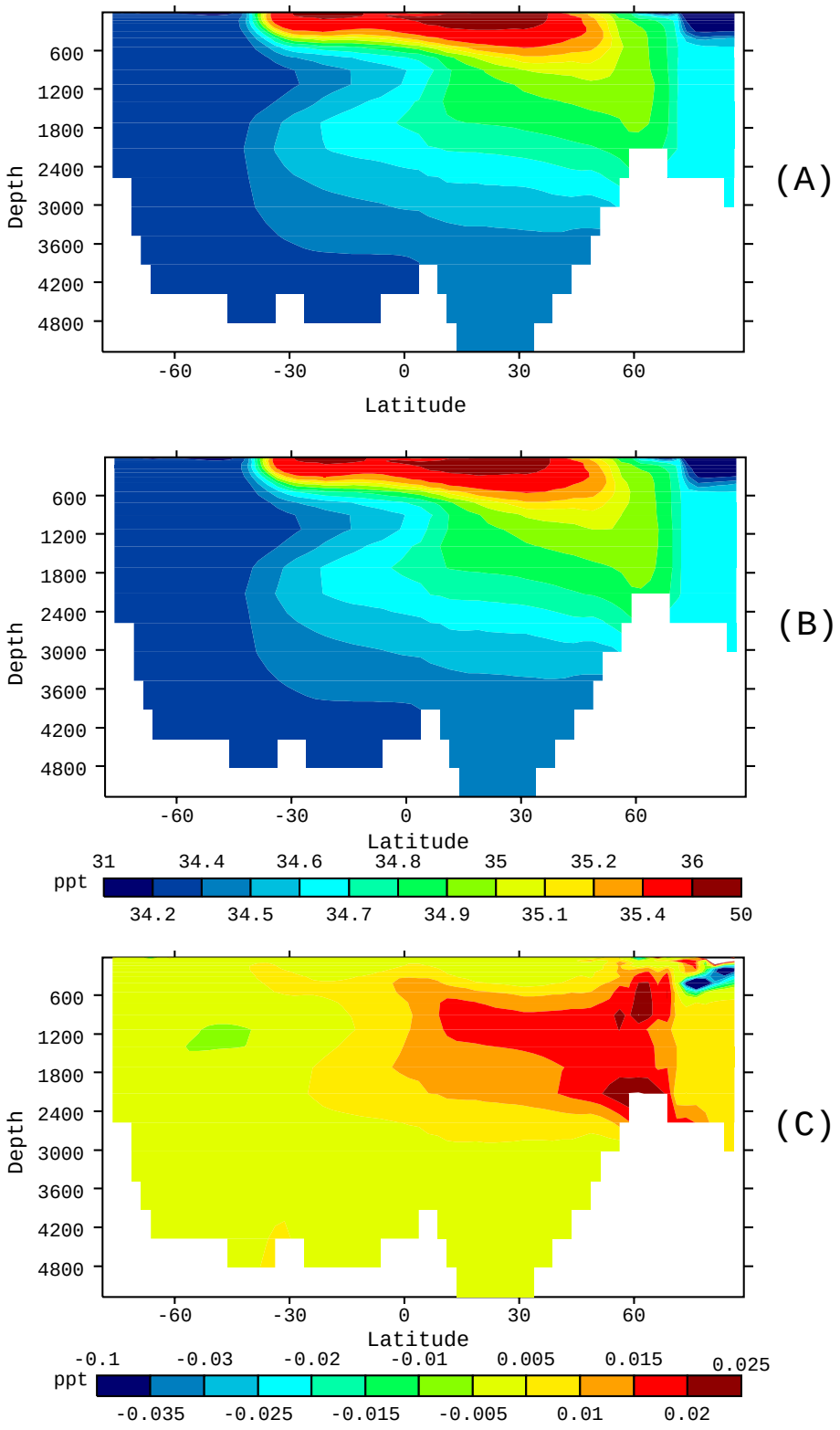

Figure 5.12: Atlantic zonal mean salinity. (A) Standard grid. (B) Reduced grid. (C) Reduced - Standard grid.

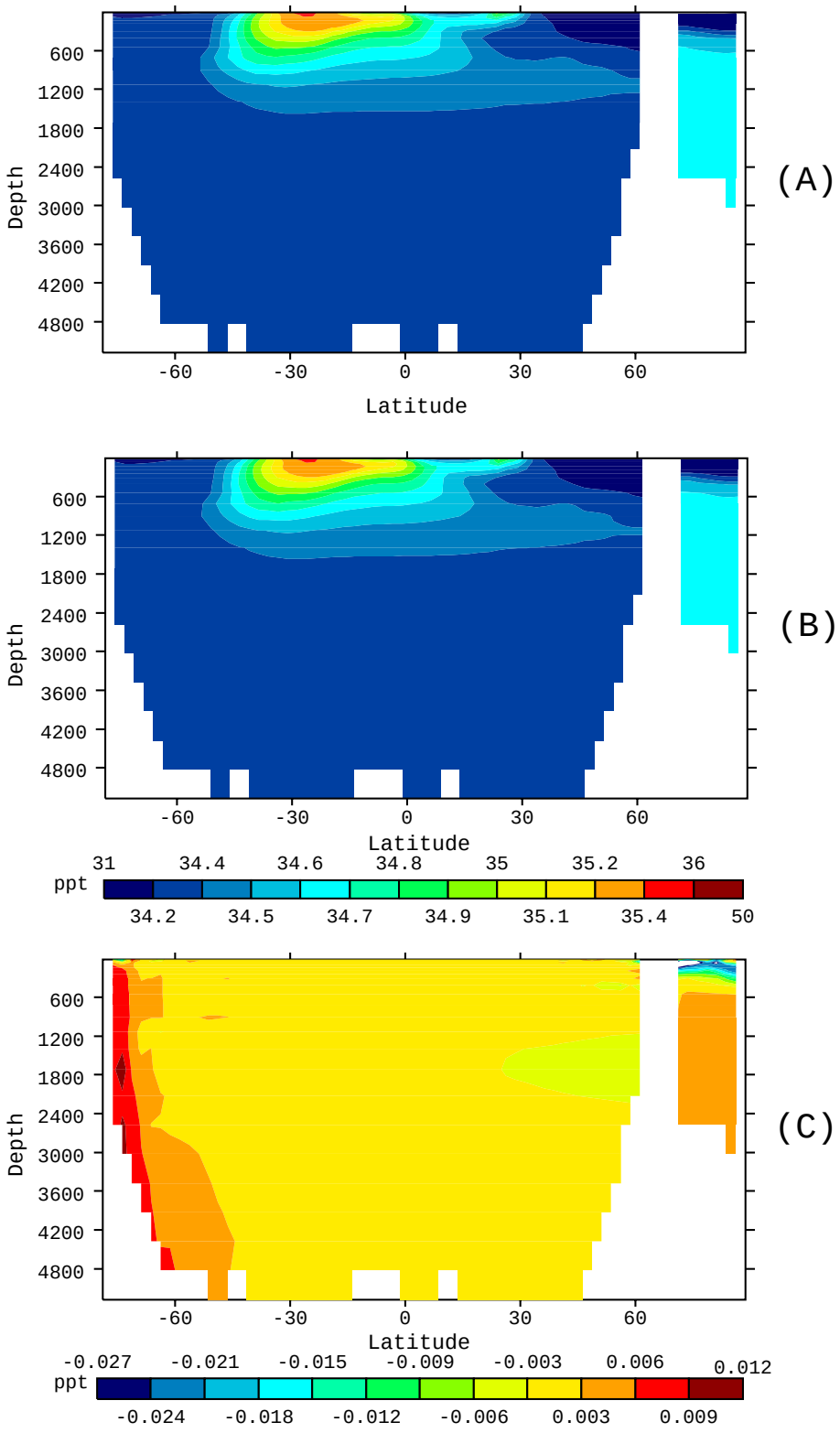

Figure 5.13: Pacific zonal mean salinity. (A) Standard grid. (B) Reduced grid. (C) Reduced - Standard grid.

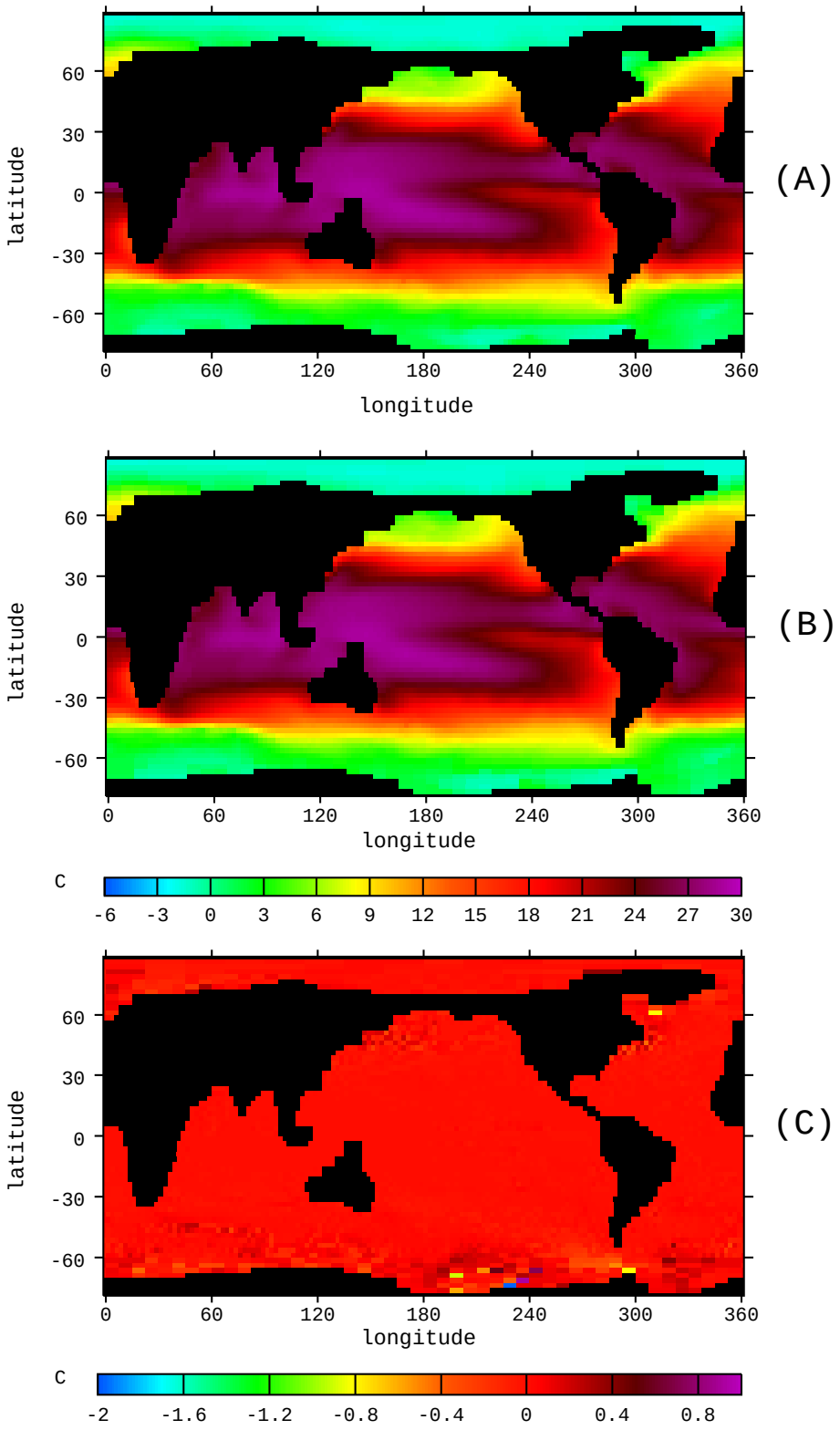

Figure 5.14: Surface temperature. (A) Standard grid. (B) Reduced grid. (C) Reduced - Standard grid.

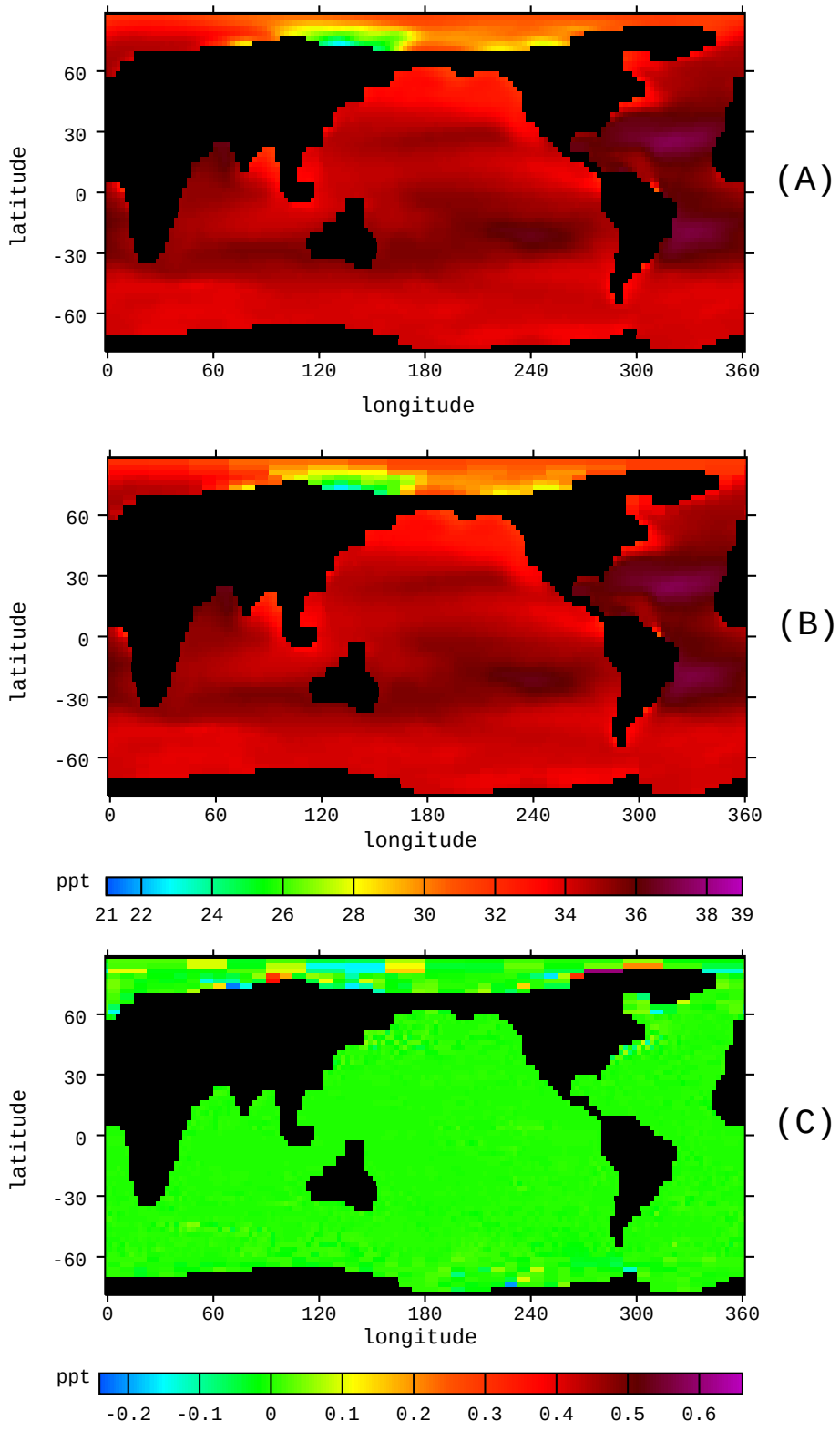

Figure 5.15: Surface salinity. (A) Standard grid. (B) Reduced grid. (C) Reduced - Standard grid.

Flow	Direction	Latitudes	Longitudes	Standard	Reduced
Drake Passage	Zonal	85 S – 50 S	70 W	139	133
South of Africa	Zonal	85 S – 30 S	20 E	140	139
Sub-Australian	Zonal	85 S – 30 S	130 E	161	159
Pacific	Meridional	32 S	120 E – 60 W	22.1	22.3

Table 5.3: Flows in the global run. Units are Sverdrups.

5.3 Surface Height

The surface heights and difference are shown in Figure 5.16. The largest differences are at the southern reduced grid interface. In this region there is a strong gradient in the height from south to north. Differences are both positive and negative with the largest of each located at and just downstream from the Drake Passage. The effect of the southern interface differences continues northward into the standard resolution region as far as 40 degrees south, though there is very little impact throughout most of the low latitudes. The largest height differences are up to a few percent in magnitude.

5.4 Circulation

As noted previously, the southern reduced grid interface is located in the middle of the strong flow known as the Antarctic circumpolar current (ACC). In this region, the lack of continental barriers allows the flow to proceed essentially uninterrupted around the whole of the Antarctic continent. The results of the limited basin runs indicated that this flow might see the most influence from the reduced grid. Table 5.3 shows the computed values of total flows at three locations of the ACC as well as one other Pacific basin location. As evident by these numbers, the reduced grid case has a 1 to 4 percent decrease in the ACC at the three locations. The meridional flow at 32 degrees south between Australia and

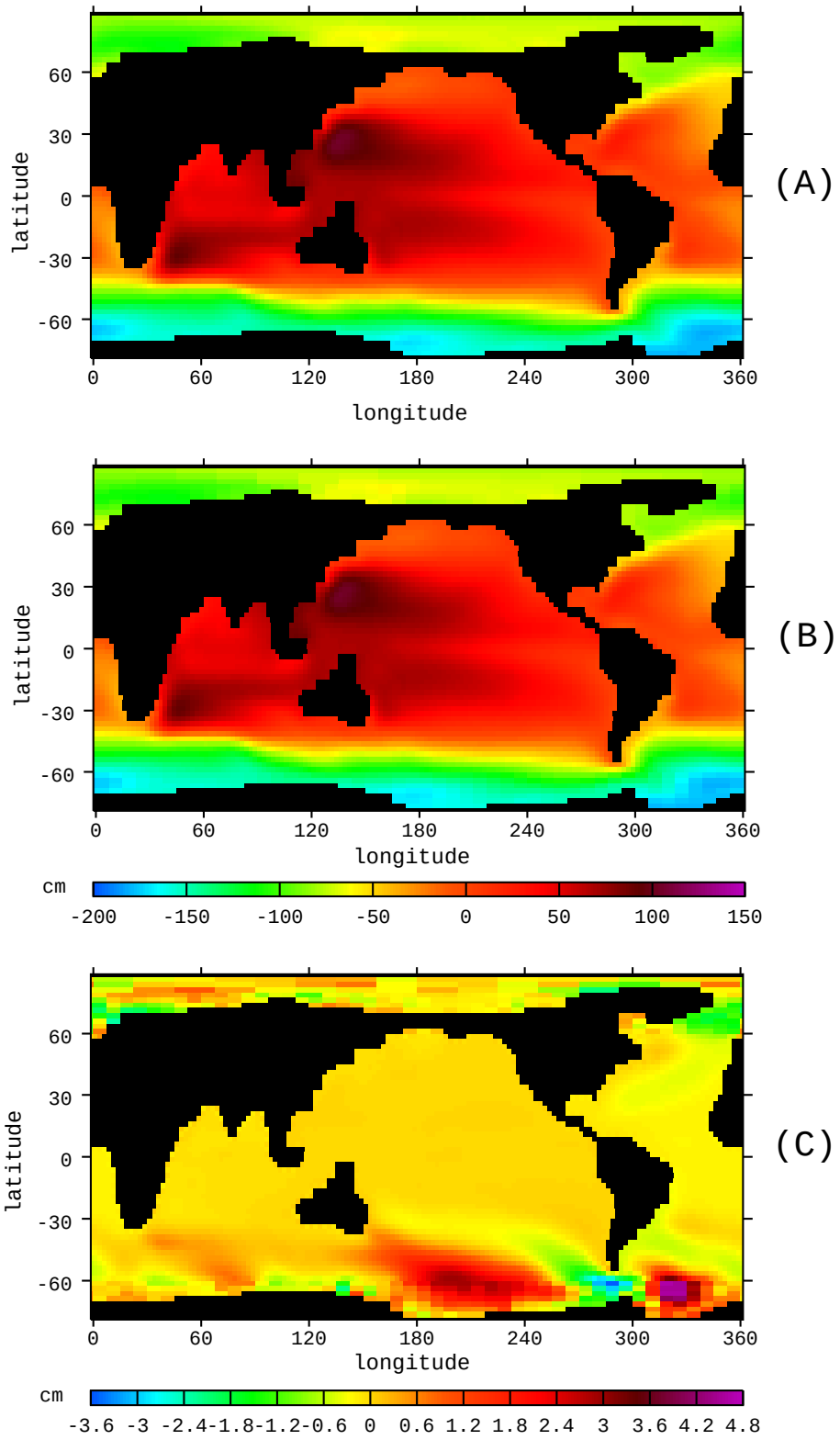

Figure 5.16: Surface height. (A) Standard grid. (B) Reduced grid. (C) Reduced - Standard grid

South America shows a difference of less than 1 percent. Note that because the Bering Strait is closed this flow is equal to the Indonesian throughflow (that between Australia and Asia) as well.

Meridional overturning is shown globally in Figure 5.17 and for the Atlantic and Indopacific basins in Figure 5.18 and Figure 5.19. Overall the overturning is very nearly the same for the standard and reduced grid cases. The differences that appear in the comparisons are not necessarily caused directly by the reduced grid interfaces and are the result less of changes in overturning strength than slight shifts in position.

A closer look at the velocities at the southern reduced grid interface is given in Figure 5.20 and Figure 5.21, showing the zonal and meridional barotropic velocities at 60 degrees south. The peak zonal velocity at the Drake Passage is evident just west of 300 degrees longitude and shows good agreement between the two cases. The slightly lower velocity of the reduced grid case can be seen between 220 and 280 degrees longitude as well as between 320 and 340 degrees longitude. The largest discrepancy appears just east of the Drake Passage, where the barotropic current turns northward. The reduced grid meridional velocity is first lower and then slightly higher than the standard grid between 310 and 340 degrees longitude. Figure 5.22 and Figure 5.23 show the averaged velocity of the first four layers, i.e. the upper 118.5 meters, also at 60 degrees south latitude. The surface flow and the barotropic flow are very similar, and the same differences pertain.

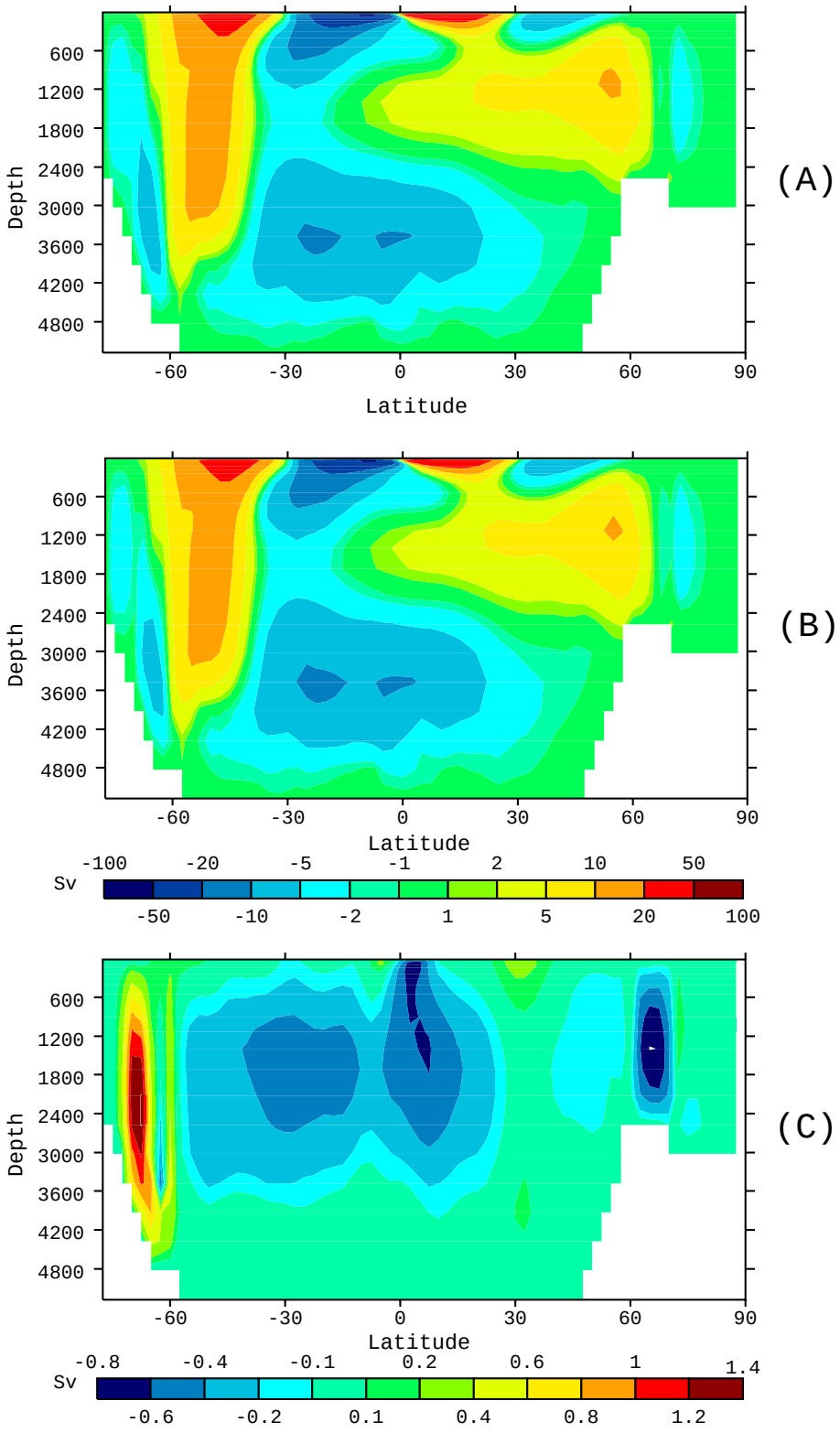

Figure 5.17: Global meridional overturning. (A) Standard grid. (B) Reduced grid. (C) Reduced - Standard grid.

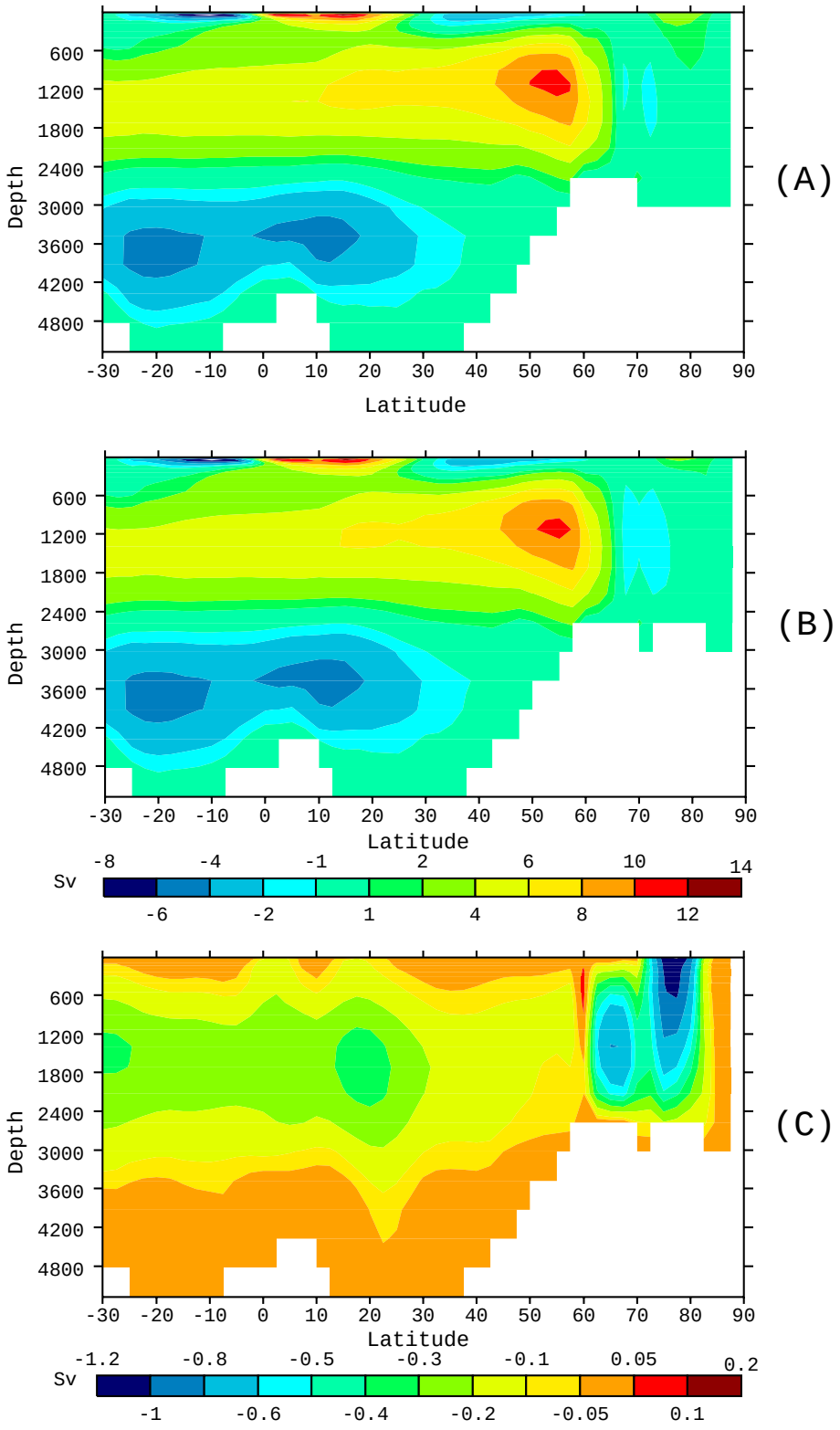

Figure 5.18: Atlantic meridional overturning. (A) Standard grid. (B) Reduced grid. (C) Reduced - Standard grid.

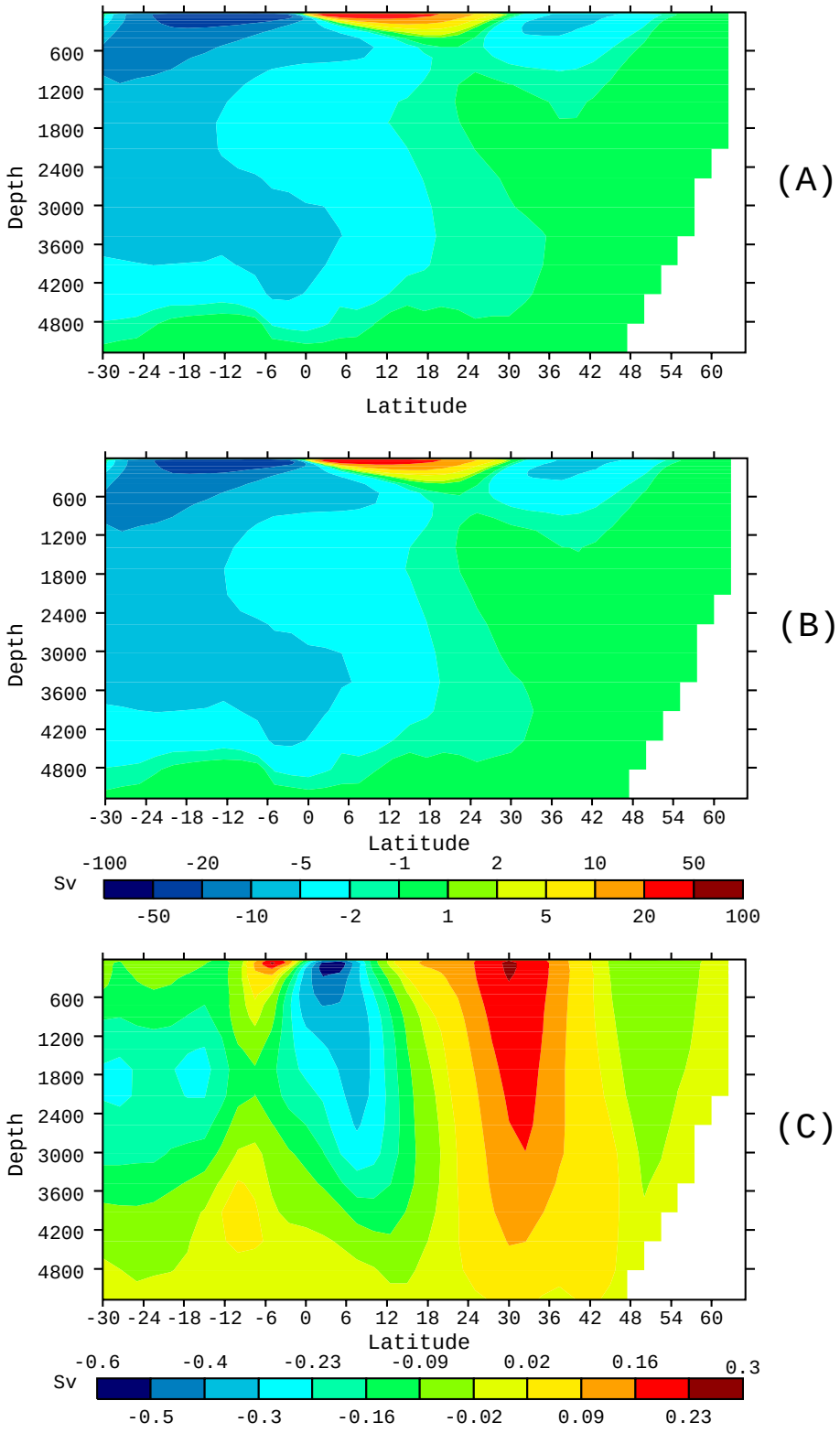

Figure 5.19: Pacific meridional overturning. (A) Standard grid. (B) Reduced grid. (C) Reduced - Standard grid.

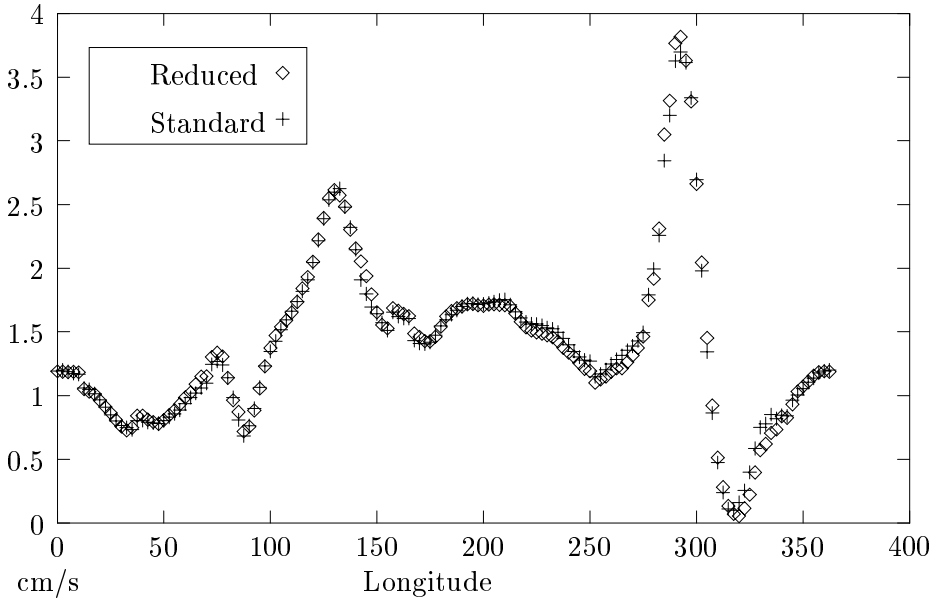

Figure 5.20: Depth averaged zonal velocity at the reduced grid interface located at 60 degrees south.

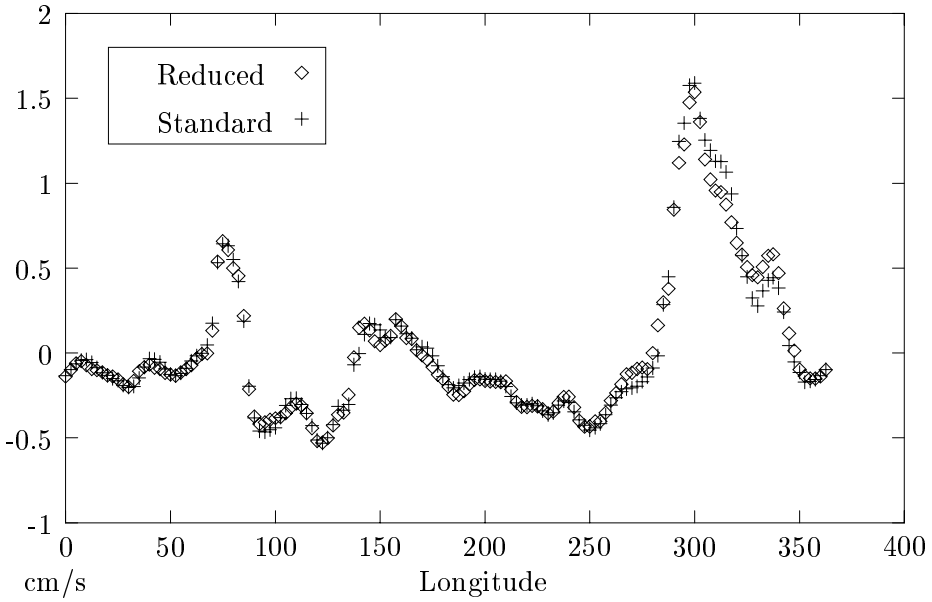

Figure 5.21: Same as Figure 5.20 except showing meridional velocity.

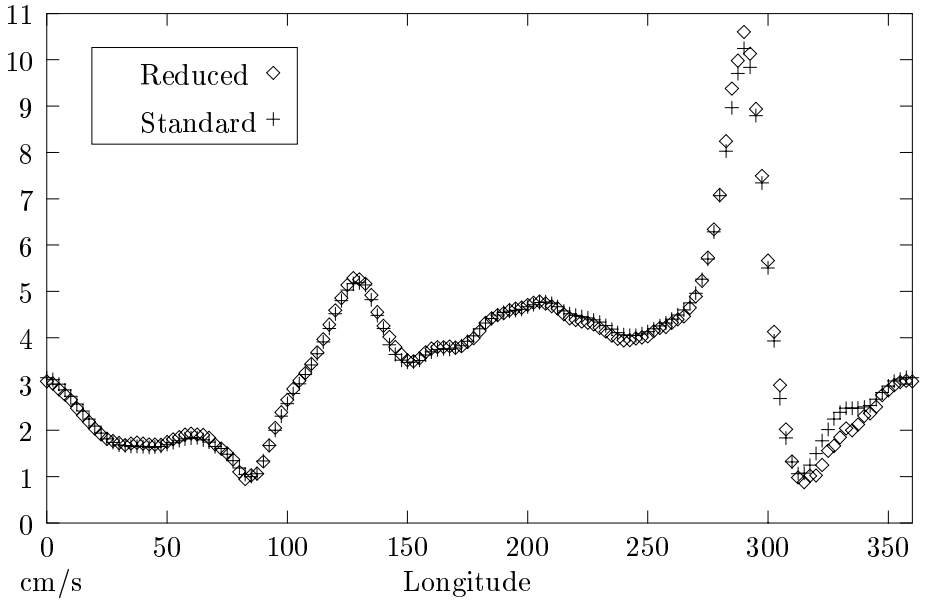

Figure 5.22: Zonal velocity averaged over the top 118.5 m at the reduced grid interface located at 60 degrees south.

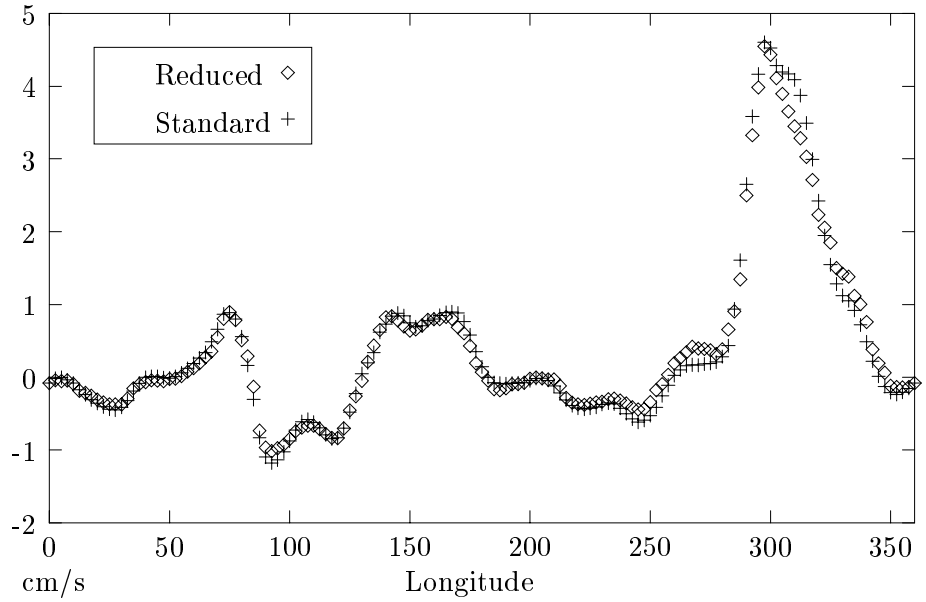

Figure 5.23: Same as Figure 5.22 except showing meridional velocity.

5.5 Parallel Timing Results

All of the simulations discussed here were performed on LLNL's ASCI Blue Pacific computer, which consists of hundreds of shared memory processor (SMP) nodes, each containing four 332 MHz PowerPC 604e processors and 1.5 GBytes of RAM. With up to two operations per clock cycle, the theoretical peak node performance is 2.656 GigaOPS. The processor to memory bandwidth is 2.1 TBytes/s, and the node to node bandwidth is 150 MBytes/s, bidirectional. There are 16 disk I/O nodes, with a bandwidth to local disk of 4.7 GBytes/s.

While this architecture supports a "mixed" programming model using shared memory parallelism on-node and message passing between nodes, the model used here uses a message passing model within nodes as well as between nodes. This approach of treating each processor as if it had its own memory and using MPI to pass messages provides for a more portable model which can run on a variety of parallel architectures. The resulting code is simpler, not having to implement two different forms of parallelism, but the additional overhead of passing messages within a node also makes it theoretically less efficient.

Timing routines in the code enable the tracking of total time, time within major parts of the code, and time spent in communication routines. The major parts of the code which are included in these results are the barotropic (surface height and barotropic velocities), baroclinic, and tracer (temperature and salinity) steps. All other parts of the code are included in the miscellaneous category. The totals in each category include communication. Additionally, the total time spent in communication is totaled on its own to get an estimate of the overhead which the parallel configuration requires.

Timing tests were made using a fixed number of steps in the same configuration as the

# PEs	Total	Barotropic	Baroclinic	Tracer	Misc.	Comm.	% Comm.
1	1980	586	574	650	168	0	0
2	1120	319	336	373	96.1	88.3	7.88
4	638	164	165	214	95.7	135	21.1
8	367	99.5	91.0	121	55.4	114	31.1
16	199	50.6	48.6	67.4	32.6	82.0	41.2
32	105	32.4	24.7	33.2	14.7	52.0	49.5
60	67.4	28.3	13.4	15.0	10.7	34.8	51.6
116	51.6	25	8.05	10.4	8.02	31.5	61.1

Table 5.4: Standard grid timing results, broken down into model components. Times are in seconds.

# PEs	Total	Barotropic	Baroclinic	Tracer	Misc.	Comm.	% Comm.
1	1630	523	445	493	172	0	0
2	885	295	228	264	98.3	29.1	3.29
4	470	172	113	132	53.1	45.2	9.62
8	260	99.3	59.1	70.7	31.2	38.7	14.9
16	149	62.3	31.1	36.0	19.6	46.2	31.0
32	93.5	46.5	17.5	18.6	10.9	39.9	42.7
61	72.9	43.0	11.7	11.7	6.54	44.9	61.6
120	66.7	41.7	7.01	9.08	8.94	46.4	69.6

Table 5.5: Reduced grid timing results, broken down into model components. Times are in seconds.

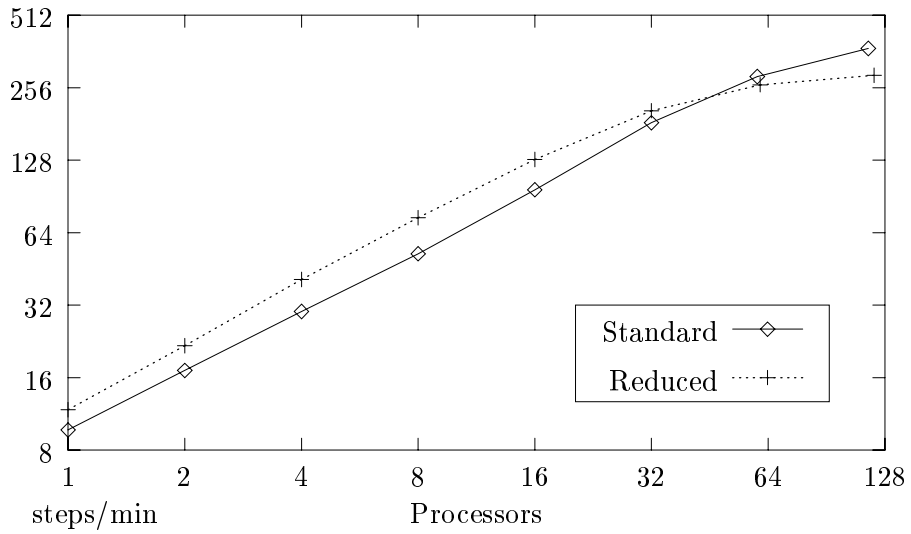

Figure 5.24: Comparison of Standard and Reduced grid parallel execution speed.

global model runs presented earlier in the chapter. Note that the number of steps is fixed, not the integration time. Therefore the difference in allowable timesteps between the two cases is *not* a factor in these tests. The raw timing numbers for both the standard and reduced cases are shown in Table 5.4 and Table 5.5 for various processor configurations. Note that for decompositions of more that 32 processors, the standard and reduced cases use slightly different numbers of processors. This is due to the difference between the decomposition restrictions of the standard and reduced model combined with the dropping of processors for subdomains which contain only land points. Refer back to Section 2.4 and Section 3.5 for details.

As can be seen in the tables, the reduced grid executes more quickly for all configurations up to 32 processors. This is shown graphically in Figure 5.24. The reasons for the slowing of the reduced grid code above 32 processors are given below. The percentage speedup of the reduced grid over the standard grid model is shown in Figure 5.25. The reduced grid is faster than the standard grid by as much as 29 percent, with an average

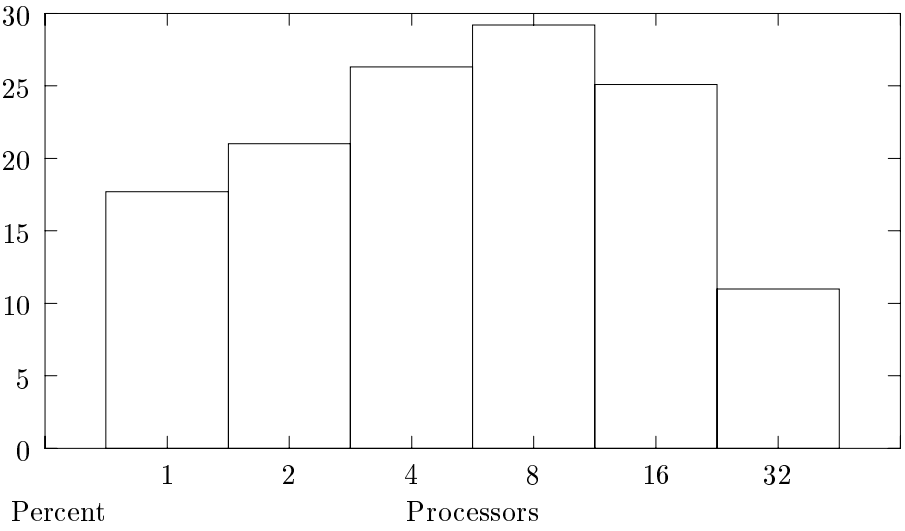

Figure 5.25: Percent speedup of the reduced grid compared to the standard grid.

speedup of 21.5 percent. In Section 5.1 it was shown that the reduced grid uses nearly 21 percent fewer cells. Though the standard grid uses a filtering step and the reduced grid requires interface calculations, the speedup of the reduced grid over the standard is about equal to the reduction in the number of cells required to cover the spherical domain. This indicates that this implementation of the reduced grid introduced no great inefficiency into the model.

To get a practical idea of just how much computer time it takes to "spin up" an ocean model to equilibrium, we can use Figure 5.24 to estimate the computer time required for a 48 processor run to execute 4000 surface tracer years. At a timestep of 6 hours, as used in the standard case here, this takes the IBM machine nearly 17 days of round-the-clock calculation. For the run described, well over a month was actually needed to achieve the 4800 year run due to practical constraints. By comparison, the reduced grid run needed one-quarter of the time due to its larger time steps.

A perfectly efficient implementation of a code on a parallel computer would see a

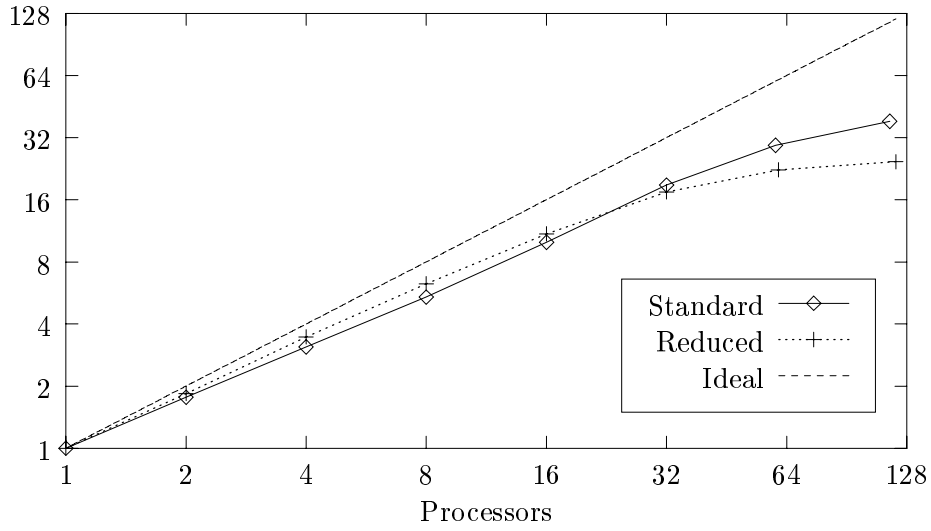

Figure 5.26: Comparison of Standard and Reduced grid parallel speedup.

linear speedup as the number of processors was increased. That is, doubling the number of processors would halve the execution time. Define the parallel speedup as

$$S_N = \frac{T_1}{T_N}, \tag{5.1}$$

where T_1 is the execution time with one processor, and T_N is the execution time with N processors. Then $S_N = N$ for perfect speedup.

A comparison of the parallel speedup for the two global cases is shown in Figure 5.26. The dashed line represents perfect speedup. The reduced case shows speedup closer to the ideal for low processor numbers but significantly poorer speedup at higher processor numbers. The poorer speedup at higher processor numbers is also the reason for the slower execution speed relative to the standard case. The relative speedup is explained by the relative coarseness of the two cases. At 60 processors the standard grid has about 163 cells per processor in the two dimensional barotropic calculations. The reduced grid has fewer than 130 cells per processor. Therefore the amount of computation compared to

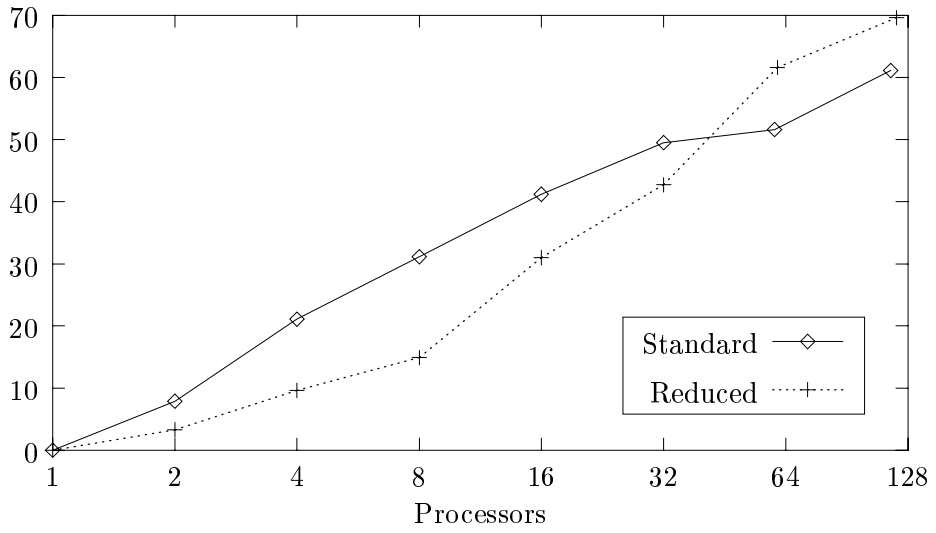

Figure 5.27: Comparison of the percentage of time used for communication as a function of the number of processors between the standard and reduced grid cases.

communication is lower for the reduced grid. This is shown in Figure 5.27.

That this is the reason for the poorer scaling is revealed in the breakdown of the timing into model components. Figure 5.28 shows the same parallel speedup of the standard case as Figure 5.26, but includes the curves for the individual model components separately. Figure 5.29 shows the same data for the reduced grid case. In both cases, the baroclinic and tracer model components show a significantly better parallel speedup than the barotropic component. The baroclinic and tracer components of the standard case show a deviation of the speedup from the ideal at low processor numbers, then fairly consistent speedup all the way up to the highest numbers. The same components of the reduced case show excellent speedup for low processor numbers that smoothly decreases. Both the standard and reduced cases have comparable speedup at high processor numbers for these components.

The barotropic component, however, shows much larger differences between the standard and reduced cases, and it is this component that contributes the most to the overall differences between model cases. In the standard case, this component follows the others

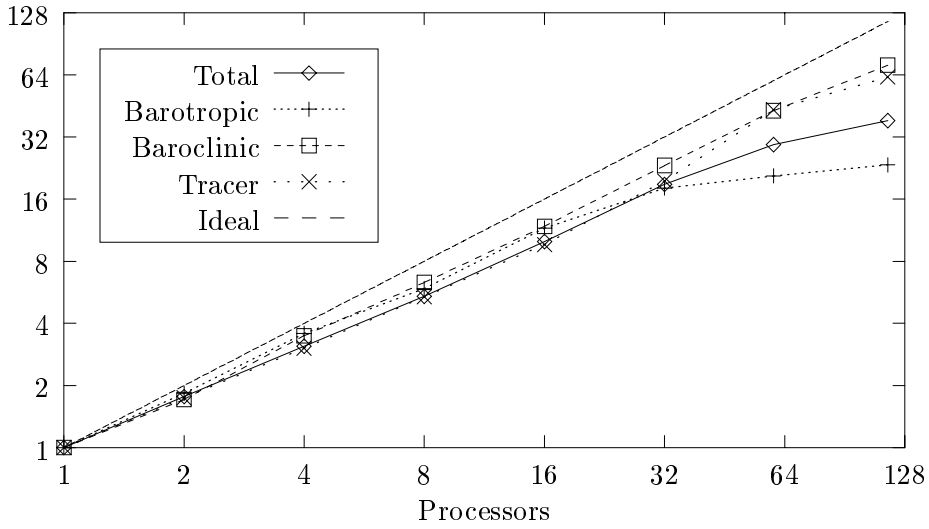

Figure 5.28: Standard grid parallel speedup, broken down into model components.

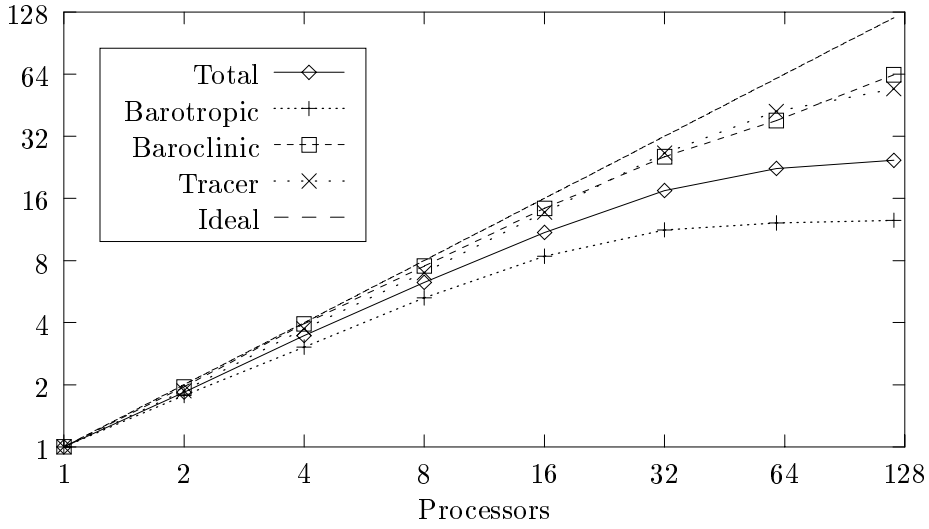

Figure 5.29: Reduced grid parallel speedup, broken down into model components.

until the processor count reaches 32, then a sudden falloff in the speedup occurs. The reduced case has smooth but further from ideal behavior throughout the test range. Both demonstrate nearly flat speedup curves at the highest numbers of processors. The main difference between the barotropic component of the model and the others is its two dimensional nature. As mentioned above, the ratio of computation to communication affects the speedup, and this component has the lowest ratio due to its lower dimensionality.

The fact that the barotropic component of the reduced grid case shows an earlier and larger decrease in the speedup is in part a result of the larger amount of communication required in this model step with the reduced grid as well as the resulting larger communication to computation ratio from the reduction in the number of cells with the reduced grid. The reduced grid requires communication during certain interface operations (see Section 3.3) which lowers the computation to communication ratio. While similar additional communication steps are required in the other model components for the reduced grid, the standard case also has extra communication required in the filtering calculations for those components. Since there is no filtering of barotropic quantities in the standard grid runs presented here, a larger discrepancy between the reduced and standard cases is seen. It is noteworthy that the addition of extra "ghost" zones at the subdomain boundaries, which are often required for more sophisticated advection schemes and the like, would eliminate the extra communication required by the reduced grid.

As mentioned in Section 5.1 runs were made with a quick implementation of filtering for the variables of the barotropic equations. As no attempt was made to make the filtering efficient for parallel runs, it is no surprise that the runs show very poor speedup. Table 5.6, Figure 5.30, and Figure 5.31 show the results of this case, which is given to demonstrate the drastic effects of the load imbalance and required communication time of a simple

# PEs	Total	Barotropic	Baroclinic	Tracer	Misc.	Comm.	% Comm.
1	2060	675	570	651	169	0	0
2	1170	365	334	371	96.2	84.3	7.21
4	1200	701	177	216	105	611	50.9
8	932	653	91.1	125	64.2	638	68.5
16	857	697	49.2	67.3	43.6	696	81.2
32	757	675	24.1	32.6	24.6	659	87.1
60	575	522	13.7	20.6	17.8	522	90.8
116	713	662	8.16	20.3	22.4	668	93.7

Table 5.6: Standard grid, barotropic filtered timing results, broken down into model components. Times are in seconds.

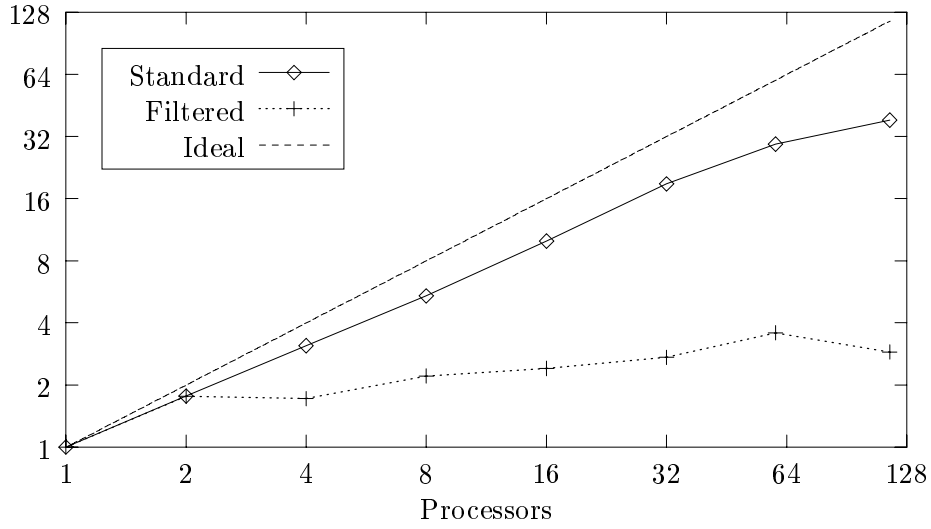

Figure 5.30: Comparison of standard and barotropic filtered grid parallel speedup.

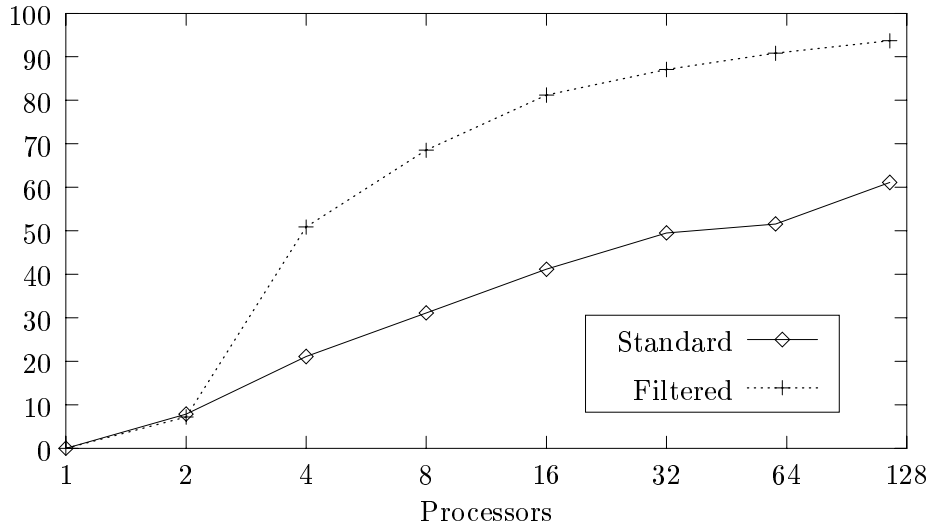

Figure 5.31: Comparison of the percentage of time used for communication as a function of the number of processors for the standard grid with and without filtering of barotropic quantities.

implementation of filtering on the speedup.

Parallel scalability refers to the effects on execution time of increasing both the number of processors and the problem size simultaneously. For example, increasing the resolution by a factor of two in all three dimensions, reducing the timestep by a factor of two, and then increasing the number of processors used by the eight would result in an identical execution time if the model had perfect scalability. The need for global communication in an algorithm is greatly detrimental to the scalability. While no tests are given here, it is noted that the explicit nature of the model numerics and the local-only communication requirements of the reduced grid method indicate a theoretically scalable model.

Chapter 6

Conclusions

6.1 Summary

This dissertation has detailed the design, implementation, and use of a reduced grid method in a parallel global ocean general circulation model. This method addressed the need for increased computational efficiency in ocean modeling in two major ways: decreasing the number of grid cells required for discretizing the sphere and increasing the length of the stability limited time step without resorting to filtering of model variables.

The reduced grid method was given in detail for the Lawrence Livermore National Laboratory's parallel ocean general circulation model. This model is very similar to many widely used models as it is based on the popular Bryan-Cox model. The method of implementation has been chosen so as to minimize the impact of the method on the existing code and to take advantage of the existing domain decomposition method for parallel processing. It has been shown that by the addition of a small number of steps using general interface operations, the existing numerical formulations can be used relatively unmodified. The interface operations are very similar to the mesh interaction of the

popular adaptive mesh refinement (AMR) method with the modifications necessary for the staggered layout of model variables used.

The resulting reduced grid model was tested for a basin of limited extent with forcing ranging from very simple wind stress to complete wind and thermohaline forcing based on observations. Two types of reduced grid interface were tested in the first runs, as well as four different types of interpolation at the interfaces. These options were narrowed down to the one that produced the best results, the "type-1" interface with linear interpolation of model variables. Comparison of results with a limited basin showed that while the influence of the interface could be seen in somewhat "noisy" velocities along the poorly resolved western boundary, the overall model result was well behaved. The surface height, overturning circulation, and temperature and salinity fields in the two halves of the domain were very near those of the corresponding constant resolution cases.

Two global runs were integrated with realistic topography and forcing to an equilibrium solution. A standard case used constant resolution combined with filtering of the baroclinic and tracer variables. A reduced grid case used a grid with four resolution regions joined by three interfaces and no filtering of model variables. Comparisons of the resulting general circulation and thermohaline fields again showed only small changes due to the reduced grid, even though the southernmost interface was purposely located in the middle of the strong Antarctic circumpolar current. The largest effect was a few percent reduction in the strength of this current at the Drake passage, though other regions saw much smaller changes in large scale flow. Tracer fields had small systematic changes, and the largest tracer changes were for individual cells due to the operation of convective adjustment over model columns of different sizes for the two cases. This change in column processes had little influence on the overall solution.

Timing results for various domain decompositions were described for LLNL's ASCI Blue Pacific computer. The reduction in the number of cells for the global domain with the reduced grid provided a corresponding reduction in the amount of computer time required per model step. The parallel speedup of the model as the number of processors increased was somewhat better than the standard case until the smaller number of cells resulted in a lower computation to communication ratio. However, for moderate numbers of processors, the larger allowable time step of the reduced grid model and the higher speed per step combine to see a dramatic — up to four times — reduction in the time to achieve model equilibrium.

6.2 Future Work

For the global run described in Chapter 5, the reduced grid method was able to achieve very similar model results with a very significant reduction in computation time. However, to see more general use in the ocean modeling field, some additional model components would need to be adapted to the reduced grid. Isopycnal mixing parameterizations, which simulate the natural mixing of model quantities along contours of constant density rather than just horizontally and vertically and have become standard in recent years, need to be included in the reduced grid model. Likewise, an ice model, which simulates the formation, melting, and movement of sea ice, would also need to be included in a production reduced grid model.

The current research also points to exciting and potentially significant future projects. Combined with what has been learned from nested grid modeling, the success of this reduced grid model for long time integrations could lead to the application of composite

meshes to the field of global ocean general circulation modeling. Higher resolution could be applied to regions which have important topological features affecting the global scale flow which normally wouldn't be resolved, and narrow boundary currents might benefit from increased resolution across their narrow widths, possibly resulting in more realistic transport of heat toward the poles. More work needs to be done to generalize the ocean code for the flexible application of these ideas, and to do so without loss of efficiency. But the constant limitations of computer power in the field of climate modeling will continue to demand such improvements in methods to get the most effective use of available resources.

Appendix A

Numerical Reference

A.1 Conservation

For long time integrations of the equations of fluid flow and ocean flow in particular, certain conservation properties are important. Quantities such as momentum, temperature, and salinity need to be conserved to prevent unphysical sources or sinks which could severely affect the solutions after long integration times. It has also been shown to be desirable to use finite difference formulations that conserve kinetic energy and the variance of temperature and salinity in the absence of dissipative terms. Arakawa first showed, in [2], that a form of "nonlinear" instability could be avoided if this conservation constraint was imposed on the energy. The following synopsis follows that of Bryan in [11].

Consider the incompressible advection equation given by

$$\frac{\partial q}{\partial t} + \nabla \cdot (\mathbf{u}\, q) = 0, \tag{A.1}$$

where $\mathbf{u}$ is the velocity and q is the advected quantity. With the domain divided into N

cells with volumes given by α_n, integrate (A.1) over each cell to give

$$\frac{\partial}{\partial t}\int q\,dV = -\oint q\,\mathbf{u}\cdot\mathbf{n}\,dS, \tag{A.2}$$

where $\mathbf{n}$ is the unit normal vector at the cell boundary. Using

$$Q_n = \frac{1}{\alpha_n}\int q_n\,dV, \tag{A.3}$$

for the average value of q_n over the cell, (A.2) becomes

$$\alpha_n\frac{\partial Q_n}{\partial t} = -\oint q\,\mathbf{u}\cdot\mathbf{n}\,dS. \tag{A.4}$$

If each cell has M faces with area and normal velocity of A_m and u_m, respectively, continuity gives

$$\sum_{m=1}^{M} u_m A_m = 0. \tag{A.5}$$

(A.4) can be approximated by

$$\alpha_n\frac{\partial Q_n}{\partial t} = -\sum_{m=1}^{M} q_m u_m A_b, \tag{A.6}$$

where q_m is the value of q at the m^{th} face. This is the flux form of the approximation used in all of the discrete forms of Section 2.3.

The total amount of the quantity q in the domain is given by

$$I_1 = \sum_{n=1}^{N} Q_n\alpha_n, \tag{A.7}$$

and the second moment by

$$I_2 = \sum_{n=1}^{N} Q_n^2\alpha_n. \tag{A.8}$$

Summing (A.6) over all of the cells in the domain yields

$$\frac{\partial}{\partial t}\sum_{n=1}^{N}\alpha_n Q_n = -\sum_{n=1}^{N}\sum_{m=1}^{M} q_m u_m A_m, \tag{A.9}$$

which can be rewritten using (A.7) to give

$$\frac{\partial I_1}{\partial t} = -\sum_{n=1}^{N}\sum_{m=1}^{M} q_m u_m A_m. \tag{A.10}$$

Since q_m, u_m, and A_m are values defined on the faces, they each occur for two cells in the summation with opposite signs for the velocity, except for the faces on the domain boundary. With the condition of no normal flow at the domain boundary, those velocities are zero. Then all of the terms in the summation cancel or are zero, and

$$\frac{\partial I_1}{\partial t} = 0. \tag{A.11}$$

So by using the flux form to approximate our equations, the total quantity will be conserved.

Similarly, multiplying (A.6) by Q_n and summing over all cells yields

$$\frac{\partial I_2}{\partial t} = -2\sum_{n=1}^{N}\sum_{m=1}^{M} q_m Q_n u_m A_m. \tag{A.12}$$

In general, this does not sum to zero. However, using the average value of q in neighboring cells as the value of q_m on the interface, i.e.

$$q_b = \frac{Q_n + Q_m}{2}, \tag{A.13}$$

then (A.12) can be written as

$$\frac{\partial I_2}{\partial t} = -\sum_{n=1}^{N}\left[Q_n^2\sum_{m=1}^{M} u_m A_m + \sum_{m=1}^{M} Q_n Q_m u_m A_m\right]. \tag{A.14}$$

The continuity equation, (A.5), shows that the first term is zero. The second term is again made of pairs of equal values with opposite signs for the velocity for interior faces and zero velocity values for boundary faces. Thus by using the average value of (A.13),

$$\frac{\partial I_2}{\partial t} = 0, \tag{A.15}$$

and second moments will be conserved. If q is momentum, then kinetic energy is conserved, and for temperature or salinity, the variance is conserved.

A.2 Accuracy

It is desirable for the solution of a finite difference method to accurately represent the solution of the differential equation it is approximating. A method is called "consistent" when the difference equation approaches the differential equation as the time and space differences, Δt and Δx, approach zero. This section gives analyses of the accuracy of methods representative of the ocean model and the reduced grid.

A.2.1 Leapfrog Time Differencing

Starting with the linear advection equation

$$u_t + a u_x = 0, \tag{A.16}$$

and discretizing with centered differences in time and space gives

$$u_j^{n+1} - u_j^{n-1} + \frac{ak}{h}\left(u_{j+1}^n - u_{j-1}^n\right) = 0, \tag{A.17}$$

where k and h are the time and space steps, respectively.

Taylor expanding terms, and writing $u(x,t)$ as u,

$$u(x,t+k) = u + k u_t + \frac{k^2}{2}u_{tt} + \frac{k^3}{6}u_{ttt} + \frac{k^4}{24}u_{tttt} + O(k^5) \tag{A.18}$$

$$u(x,t-k) = u - k u_t + \frac{k^2}{2}u_{tt} - \frac{k^3}{6}u_{ttt} + \frac{k^4}{24}u_{tttt} + O(k^5) \tag{A.19}$$

$$u(x+h,t) = u + h u_x + \frac{h^2}{2}u_{xx} + \frac{h^3}{6}u_{xxx} + \frac{h^4}{24}u_{xxxx} + O(h^5) \tag{A.20}$$

$$u(x-h,t) = u - h u_x + \frac{h^2}{2}u_{xx} - \frac{h^3}{6}u_{xxx} + \frac{h^4}{24}u_{xxxx} + O(h^5). \tag{A.21}$$

Then these expressions can be substituted into (A.17) to find the truncation error, τ.

$$2u_t + \frac{k^2}{3}u_{ttt} + O(k^4) + \frac{a}{h}\left[h u_x + \frac{h^3}{3}u_{xxx} + O(h^5)\right] = \tau. \tag{A.22}$$

Making use of the advection equation itself, the first terms on each side cancel, leaving

$$\tau = \frac{k^2}{3} u_{ttt} + \frac{ah^2}{3} u_{xxx} + O(h^4). \tag{A.23}$$

Again using (A.16) and calculating multiple mixed derivatives,

$$u_{xxt} + a u_{xxx} = u_{xtt} + a u_{xxt} = u_{ttt} + a u_{xtt} = 0, \tag{A.24}$$

allows the expression of the time derivative in (A.22) in terms of a spatial derivative,

$$u_{ttt} = -a u_{xtt} = a^2 u_{xxt} = -a^3 u_{xxx}, \tag{A.25}$$

giving

$$\begin{aligned} \tau &= -\frac{k^2 a^3}{3} u_{xxx} + \frac{ah^2}{3} u_{xxx} + O(h^4) \\ &= \frac{k^2 a^3}{3} u_{xxx} \left(\frac{h^2}{k^2 a^2} - 1 \right) + O(h^4). \end{aligned} \tag{A.26}$$

So it can be seen that the method is second order accurate in space and time, with the largest error term being a dispersive term.

A.2.2 Euler Backward Time Differencing

Again discretizing the linear advection equation, (A.16), using the two-step Euler backward method, described in Section 2.3.9, with centered differences in space gives

$$u_m^* = u_m^n - \frac{ak}{2h} \left(u_{m+1}^n - u_{m-1}^n \right) \tag{A.27}$$

$$u_m^{n+1} = u_m^n - \frac{ak}{2h} \left(u_{m+1}^* - u_{m-1}^* \right). \tag{A.28}$$

Substituting (A.27) into (A.28) yields

$$u_m^{n+1} = u_m^n - \frac{ak}{2h} \left(u_{m+1}^n - u_{m-1}^n \right) + \left(\frac{ak}{2h} \right)^2 \left(u_{m+2}^n - 2u_m^n + u_{m-2}^n \right). \tag{A.29}$$

Expanding in a Taylor series about u_m^n,

$$\begin{aligned} u + ku_t + \frac{k^2}{2}u_{tt} + \mathrm{O}\left(k^3\right) = \\ u - \frac{ak}{2h}\left(2hu_x + \mathrm{O}\left(h^3\right)\right) + \left(\frac{ak}{2h}\right)^2 \left[(2h)^2 u_{xx} + \mathrm{O}\left(h^4\right)\right], \end{aligned} \tag{A.30}$$

which simplifies to

$$u_t + \frac{k}{2}u_{tt} + \mathrm{O}\left(k^2\right) = -au_x + a^2 k u_{xx} + \mathrm{O}\left(h^2\right). \tag{A.31}$$

Again using the differentiating the advection equation,

$$u_t t = -au_{xt} = a^2 u_{xx}, \tag{A.32}$$

allowing (A.31) to be written as

$$u_t = -au_x + \frac{a^2 k}{2}u_{xx} + \mathrm{O}\left(k^2, h^2\right). \tag{A.33}$$

Thus the Euler backward method is first-order accurate in time, with an effective diffusivity of $a^2 k/2$ which decreases as the time step is reduced.

A.2.3 Reduced Grid Interface

Calculating a first derivative in the y-direction on the fine side of the Cartesian-coordinate interface shown in Figure A.1 gives

$$\delta_y u_{6,2} = \frac{\frac{1}{2}\left(\tilde{u}_{6,3} + u_{6,2}\right) - \frac{1}{2}\left(u_{6,2} + u_{6,1}\right)}{h_2}, \tag{A.34}$$

where $\tilde{u}$ is a value obtained by interpolation. The interpolation formulas of Section A.4 allow writing this value in terms of the true value, e.g.

$$\tilde{u}_{6,3} = u_{6,3} + \mathrm{O}\left(h_1^I\right), \tag{A.35}$$

where I is the order of accuracy of the interpolation method. This gives

$$\delta_y u_{6,2} = \frac{\frac{1}{2}\left(u_{6,3} + u_{6,2}\right) - \frac{1}{2}\left(u_{6,2} + u_{6,1}\right)}{h_2} + \mathrm{O}\left(\frac{h_1^I}{h_2}\right). \tag{A.36}$$

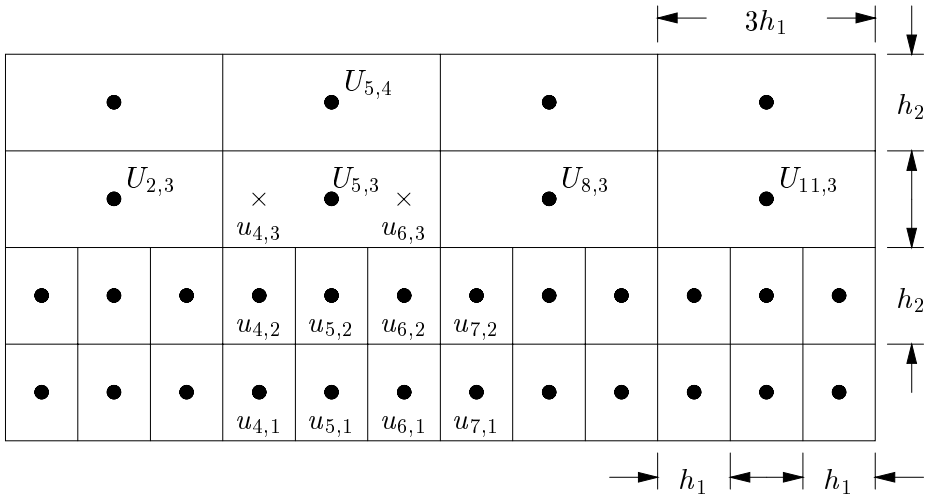

Figure A.1: An Interface

Expanding each term about $u_{6,2}$ as done above in the example of centered differences gives

$$\delta_y u_{6,2} = u_y + \mathrm{O}\left(h_2^2, \frac{h_1^I}{h_2}\right). \tag{A.37}$$

Similarly, calculating a second derivative in the y-direction on the fine side of the interface gives

$$\delta_{yy} u_{6,2} = \frac{(\tilde{u}_{6,3} - u_{6,2}) - (u_{6,2} - u_{6,1})}{h_2^2} \tag{A.38}$$

$$= \frac{(u_{6,3} - u_{6,2}) - (u_{6,2} - u_{6,1})}{h_2^2} + \mathrm{O}\left(\frac{h_1^I}{h_2^2}\right) \tag{A.39}$$

$$= u_{yy} + \mathrm{O}\left(h_2^2, \frac{h_1^I}{h_2^2}\right). \tag{A.40}$$

On the coarse side of the interface, the fluxes used at the southern face are given by the average of the fine fluxes at the interface. So the first derivative is given by

$$\delta_y U_{5,3} = \frac{\frac{1}{2}(U_{5,4} + U_{5,3}) - \frac{1}{3}\left[\frac{1}{2}(\tilde{u}_{4,3} + u_{4,2}) + \frac{1}{2}(U_{5,3} + u_{5,2}) + \frac{1}{2}(\tilde{u}_{6,3} + u_{6,2})\right]}{h_2}. \tag{A.41}$$

Again using interpolated values for $\tilde{u}_{4,3}$ and $\tilde{u}_{6,3}$ of order h_1^I, this becomes

$$\delta_y U_{5,3} = \frac{1}{6h_2}(3U_{5,4} + 2U_{5,3} - u_{4,3} - u_{6,3} - u_{4,2} - u_{5,2} - u_{6,2}) + \mathrm{O}\left(\frac{h_1^I}{h_2}\right). \tag{A.42}$$

Expanding terms about $U_{5,3}$ gives

$$\delta_y U_{5,3} = u_y - \frac{h_1^2}{2h_2} u_{xx} + \mathrm{O}\left(h_2^2, \frac{h_1^I}{h_2}\right). \tag{A.43}$$

Similarly, for the second derivative on the coarse side of the interface,

$$\delta_{yy} U_{5,3} = \frac{(U_{5,4} - U_{5,3}) - \frac{1}{3}\left[(\tilde{u}_{4,3} - u_{4,2}) + (U_{5,3} - u_{5,2}) + (\tilde{u}_{6,3} - u_{6,2})\right]}{h_2^2} \tag{A.44}$$

$$= \frac{1}{3h_2^2}\left(3U_{5,4} - 4U_{5,3} - u_{4,3} + u_{4,2} + u_{5,2} - u_{6,3} + u_{6,2}\right) + \mathrm{O}\left(\frac{h_1^I}{h_2^2}\right) \tag{A.45}$$

$$= u_{yy} - \frac{h_1^2}{h_2} u_{xxy} + \mathrm{O}\left(h_2^2, \frac{h_1^I}{h_2^2}\right). \tag{A.46}$$

These results show that the interface numerics will be unconditionally consistent with third order accuracy of the interpolation formula and conditionally consistent with second order accuracy. That is, with linear interpolation, consistency strictly requires that h_1 go to zero faster that h_2. The use of no interpolation to preserve second moments, as detailed in Section A.1, leads to an inconsistent scheme at the interface and potentially large errors. This is in fact what is seen in the limited basin tests of Section 4.1. The truncation error on the coarse side of the interface shows that regardless of the interpolation scheme used, there will be local errors of less than second order in both the advective and diffusive terms.

A.3 Stability

It is obviously desirable for the solution of a finite difference method to approach the solution of the differential equation it is approximating. Therefore, if the solution to differential equation remains bounded for all time, the difference equation should also remain bounded as it is integrated forward. A method is called "unstable" if there exist initial solutions for which the finite difference solution becomes unbounded as the time

level goes to infinity. This section will demostrate stability for linearized example equations comparable to those used in the ocean model.

A.3.1 Leapfrog Method for Advection

Looking now at the advection terms, the one-way wave equation,

$$\frac{\partial u}{\partial t} + c\frac{\partial u}{\partial x} = 0, \tag{A.47}$$

is discretized with a leapfrog in time, centered in space method to yield

$$\frac{v_m^{n+1} - v_m^{n-1}}{2k} + c\frac{v_{m+1}^n - v_{m-1}^n}{2h} = 0, \tag{A.48}$$

or

$$v_m^{n+1} = v_m^{n-1} - cr(v_{m+1}^n - v_{m-1}^n), \tag{A.49}$$

where k and h are the time step and spatial grid size, respectively, $r = k/h$, and superscripts and subscripts indicate time level and grid index, respectively.

The initial condition can be expressed as a Fourier series,

$$v_m^0 = \sum_{n=0}^{N} \mathbf{A}_n e^{ib_n mh}, \qquad m = 0, 1, \ldots, N, \tag{A.50}$$

with $b_n = n\pi/Nh$ and $Nh = L$. Since (A.47) is linear, only the propagation of one Fourier component need be considered, since the individual components may be superimposed.

As time increases, the propagation of this term may be written as

$$v_m^n = e^{ibmh}e^{\alpha nk} = e^{ibmh}g^n, \tag{A.51}$$

where $g = e^{\alpha k}$ and α is, in general, a complex constant. g is known as the amplification factor.

Since the exact solution of (A.47) does not grow in time, the finite difference equation, (A.49), will be stable if $|u_m^n|$ remains bounded for all $n \leq J$ as h and k go to zero. A necessary and sufficient condition for this is

$$|g| \leq 1. \tag{A.52}$$

Substituting (A.51) into (A.49) gives

$$e^{ibmh} g^{n+1} = e^{ibmh} g^{n-1} - cr[e^{ib(m+1)h} g^n - e^{ib(m-1)h} g^n]. \tag{A.53}$$

Dividing by e^{ibmh} gives

$$\begin{aligned} g^{n+1} &= g^{n-1} - crg^n(e^{ibh} - e^{-ibh}) \\ &= g^{n-1} - 2icr\sin(bh)g^n, \end{aligned} \tag{A.54}$$

which, with the addition of the expression

$$g^n = (1)g^n + (0)g^{n-1}, \tag{A.55}$$

can be written as a system of equations

$$\begin{pmatrix} g^{n+1} \\ \\ g^n \end{pmatrix} = G \begin{pmatrix} g^n \\ \\ g^{n-1} \end{pmatrix}, \tag{A.56}$$

where

$$G = \begin{pmatrix} -2icr\sin(bh) & 1 \\ \\ 1 & 0 \end{pmatrix}. \tag{A.57}$$

For a multistep problem, the amplification factor is replaced by an amplification matrix, and likewise the stability criterion becomes

$$\max|\lambda_i| \leq 1, \tag{A.58}$$

where the λ_i are the eigenvalues of the matrix G.

The eigenvalues are given in this case by

$$\lambda_{\pm} = -icr\sin(bh) \pm \sqrt{1 - c^2 r^2 \sin^2 bh}. \tag{A.59}$$

If $c^2 r^2 \sin^2 bh > 1$, the square root term is imaginary, and $|\lambda_{\pm}| > 1$, indicating instability. When $c^2 r^2 \sin^2 bh \leq 1$, which in general requires $cr \leq 1$, then

$$\begin{aligned} |\lambda_{\pm}|^2 &= c^2 r^2 \sin^2 bh + (1 - c^2 r^2 \sin^2 bh) \\ &= 1. \end{aligned} \tag{A.60}$$

Thus the stability condition is

$$k \leq \frac{h}{c}. \tag{A.61}$$

Note that as long as the stability condition is met, (A.60) shows that the damping factor will always be one. Thus no modes are damped from the solution. Often some damping of selected modes is desired, in which case other time stepping schemes are used (see Euler Backward below).

A.3.2 Euler Backward Method for Advection

As the mixing timesteps in the model use a backward Euler timestepping scheme (see Section 2.3.9), the stability of that method for an example equation will be given here. Again using the linear advection equation, (A.47), along with centered differences, the Euler backward method described in Section 2.3.9 in discrete form is given by

$$u_m^{n'} = u_m^n - \frac{ck}{2h}\left(u_{m+1}^n - u_{m-1}^n\right) \tag{A.62}$$

$$u_m^{n+1} = u_m^n - \frac{ck}{2h}\left(u_{m+1}^{n'} - u_{m-1}^{n'}\right), \tag{A.63}$$

where k is the timestep and h is the grid spacing. Substituting $u_m^n = g^n e^{ibmh}$ into (A.62) gives

$$u_m^{n'} = \left[1 - \frac{ck}{h} i \sin(bh)\right] u_m^n. \tag{A.64}$$

This is then substituted into (A.63) to give

$$u_m^{n+1} = u_m^n - \frac{ck}{2h}\left[1 - \frac{ck}{h}\sin(bh)\right]\left(u_{m+1}^n - u_{m-1}^n\right), \tag{A.65}$$

yielding

$$g = 1 - i\alpha(1 - i\alpha) = 1 - i\alpha - \alpha^2, \tag{A.66}$$

where $\alpha = \frac{ck}{h}\sin(bh)$. The method will be stable if $|g| \leq 1$, i.e. if

$$\sqrt{\alpha^4 - \alpha^2 + 1} \leq 1, \tag{A.67}$$

which is satisfied if $|\alpha| \leq 1$, giving the stability condition

$$k \leq \frac{h}{c}. \tag{A.68}$$

This method is sometimes referred to as a Matsuno scheme in meteorology. The damping factor shows that this scheme will damp in a wave-number dependent manner when the equality in (A.68) does not hold. For a wavelength of $2h$, there is no damping, and as wavelength increases the damping diminishes.

A.3.3 Leapfrog Method for Diffusion

Now turning to the stability of the diffusion equation

$$\frac{\partial u}{\partial t} = a\frac{\partial^2 u}{\partial x^2}, \tag{A.69}$$

use the same leapfrog in time method to get

$$\frac{v_m^{n+1} - v_m^{n-1}}{2k} = a\frac{v_{m+1}^n - 2v_m^n + v_{m-1}^n}{h^2}, \tag{A.70}$$

or

$$v_m^{n+1} = v_m^{n-1} + as(v_{m+1}^n - 2v_m^n + v_{m-1}^n), \tag{A.71}$$

where $s = 2k/h^2$. Substituting (A.51) into (A.71) gives

$$e^{ibmh}g^{n+1} = e^{ibmh}g^{n-1} + as[e^{ib(m+1)h}g^n - 2e^{ibmh}g^n + e^{ib(m-1)h}g^n]. \tag{A.72}$$

Dividing by e^{ibmh} gives

$$\begin{aligned} g^{n+1} &= g^{n-1} + asg^n(e^{ibh} - 2 + e^{-ibh}) \\ &= g^{n-1} + 4as\sin^2(bh/2)g^n. \end{aligned} \tag{A.73}$$

Again, the additional expression

$$g^n = (1)g^n + (0)g^{n-1}, \tag{A.74}$$

may be used to write a system of equations

$$\begin{pmatrix} g^{n+1} \\ \\ g^n \end{pmatrix} = G \begin{pmatrix} g^n \\ \\ g^{n-1} \end{pmatrix}, \tag{A.75}$$

where

$$G = \begin{pmatrix} 4as\sin^2(bh/2) & 1 \\ \\ 1 & 0 \end{pmatrix}. \tag{A.76}$$

The eigenvalues of this system are

$$\lambda_\pm = 2as\sin^2(bh/2) \pm \sqrt{4a^2s^2\sin^4(bh/2) + 1}, \tag{A.77}$$

so

$$|\lambda_\pm|^2 = 1 + 8a^2s^2\sin^2(bh/2) \pm 4as\sin^2(bh/2)\sqrt{4a^2s^2\sin^4(bh/2)+1}. \tag{A.78}$$

Thus one of the eigenvalues will always be greater than one, so the method is unconditionally unstable.

A.3.4 Leapfrog Method for Advection-Diffusion

Proceeding toward the ocean model equations, which here are simplified to a combined advection-diffusion equation,

$$\frac{\partial u}{\partial t} + c\frac{\partial u}{\partial x} = a\frac{\partial^2 u}{\partial x^2}, \tag{A.79}$$

the leapfrog method can be used with the diffusion terms lagged in time, i.e.

$$u_m^{n+1} = u_m^{n-1} - cr(u_{m+1}^n - u_{m-1}^n) + as(u_{m+1}^{n-1} - 2u_m^{n-1} + u_{m-1}^{n-1}). \tag{A.80}$$

This is essentially the leapfrog method for the advection terms combined with a forward-time method for the diffusion terms with a time step of $2k$.

Substituting (A.51) into (A.80) and dividing by e^{ibmh} gives

$$\begin{aligned} g^{n+1} &= g^{n-1} - 2icr\sin(bh)g^n - 4as\sin^2(bh/2)g^{n-1} \\ &= -2icr\sin(bh)g^n + [1 - 4as\sin^2(bh/2)]g^{n-1}. \end{aligned} \tag{A.81}$$

Thus resulting in the system

$$\begin{pmatrix} g^{n+1} \\ \\ g^n \end{pmatrix} = G \begin{pmatrix} g^n \\ \\ g^{n-1} \end{pmatrix}, \tag{A.82}$$

where

$$G = \begin{pmatrix} -2icr\sin(bh) & 1 - 4as\sin^2(bh/2) \\ \\ 1 & 0 \end{pmatrix}, \tag{A.83}$$

with eigenvalues

$$\lambda_{\pm} = -icr\sin(bh) \pm \sqrt{1 - 4as\sin^2(bh/2) - c^2r^2\sin^2 bh}. \tag{A.84}$$

If $c^2r^2\sin^2 bh > 1$ then the square root term is imaginary, and $|\lambda_-| > 1$. If $c^2r^2\sin^2 bh \leq 1$, which in general requires $cr \leq 1$, then the square root term will be real if

$$4as\sin^2(bh/2) + c^2r^2\sin^2 bh \leq 1, \tag{A.85}$$

which is, in general, true if

$$4as + c^2r^2 = \frac{8ka + c^2k^2}{h^2} \leq 1. \tag{A.86}$$

This results in

$$|\lambda| = 1 - 4as\sin^2(bh/2), \tag{A.87}$$

and stability requires that $2as \leq 1$, or

$$k \leq \frac{h^2}{4a}. \tag{A.88}$$

So the requirements for stability include (A.61) and (A.88) which are the requirements for the advection and diffusion terms individually, with the additional requirement of (A.86). But note that if

$$\frac{h^2}{8a} \leq k \leq \frac{h^2}{4a}, \tag{A.89}$$

then (A.88) is satisfied, but (A.86) is not. The stability condition of (A.86) is more restrictive than the individual conditions for advection and diffusion considered separately, (A.61) and (A.88).

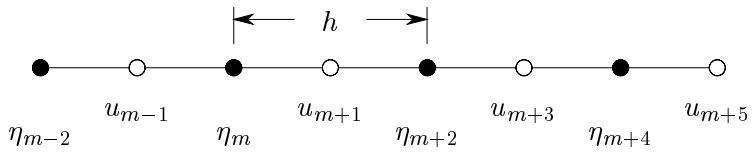

Figure A.2: Staggered grid for shallow-water equations

A.3.5 Shallow-Water Equations

Consider the one-dimensional linearized shallow-water equations

$$\frac{\partial u}{\partial t} = -g\frac{\partial \eta}{\partial x} \tag{A.90}$$

$$\frac{\partial \eta}{\partial t} = -H\frac{\partial u}{\partial x}, \tag{A.91}$$

where H is the depth of the fluid. Using a staggered arrangement of the variables u and η, shown in Figure A.2, along with centered differences gives

$$\frac{u_m^{n+1} - u_m^{n-1}}{2k} = -g\frac{\eta_{m+1}^n - \eta_{m-1}^n}{h} \tag{A.92}$$

$$\frac{\eta_m^{n+1} - \eta_m^{n-1}}{2k} = -H\frac{u_{m+1}^n - u_{m-1}^n}{h}, \tag{A.93}$$

where k is the timestep and h is the grid spacing. Substituting into (A.92) and (A.93) a solution of the form

$$u = A^n e^{ibmh} \tag{A.94}$$

$$\eta = B^n e^{ibmh}, \tag{A.95}$$

gives the equations

$$B^{n+1} = B^{n-1} - \frac{4ikH}{h}\sin(bh)A^n \tag{A.96}$$

$$A^{n+1} = A^{n-1} - \frac{4ikg}{h}\sin(bh)B^n. \tag{A.97}$$

These equations can be written in matrix form as

$$\begin{pmatrix} A^{n+1} \\ A^{n} \\ B^{n+1} \\ B^{n} \end{pmatrix} = G \begin{pmatrix} A^{n} \\ A^{n-1} \\ B^{n} \\ B^{n-1} \end{pmatrix}, \tag{A.98}$$

where the amplification matrix is given by

$$G = \begin{pmatrix} 0 & 1 & -\frac{4ikg}{h}\sin(bh) & 0 \\ 1 & 0 & 0 & 0 \\ -\frac{4ikH}{h}\sin(bh) & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}. \tag{A.99}$$

The eigenvalues of the amplification matrix are

$$\lambda^4 - 2\left(1-\alpha\right)\lambda^2 + 1 = 0, \tag{A.100}$$

where $\alpha = \frac{8gHk^2}{h^2}\sin(bh)$, or

$$\lambda^2 = 1 - \alpha \pm \sqrt{\left[\alpha\left(\alpha - 2\right)\right]}. \tag{A.101}$$

If $\alpha > 2$ then there will be an eigenvalue with magnitude greater than one. When $\alpha \leq 2$, the root is imaginary and

$$|\lambda^2| = \sqrt{2\alpha^2 - 4\alpha + 1}. \tag{A.102}$$

Since (A.102) is less than or equal to one if $\alpha \leq 2$, the eigenvalues will be less than or equal to one if $\alpha \leq 2$, i.e. if

$$k \leq \frac{h}{2\sqrt{gH}}. \tag{A.103}$$

This stability condition is analogous to the CFL condition of the advection equation above with a characteristic velocity of surface gravity waves is $\sqrt{gH}$, and the factor of two arising from the staggered arrangement of variables.

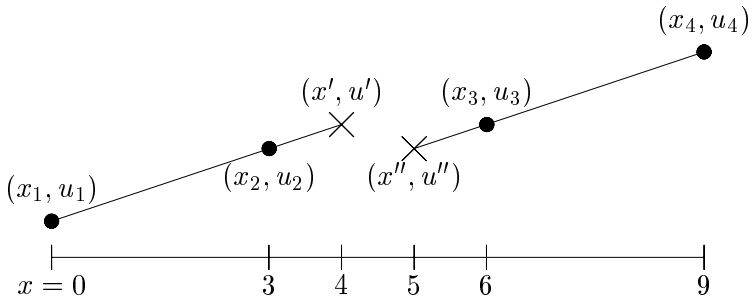

Figure A.3: Interpolation geometry

A.4 Interpolation

Section 3.3.4 discussed the general methods of interpolation for use in the reduced grid model. The source points and destination points and the influence of topography and the grid stagger were outlined. However, the actual formulas for obtaining the interpolated values once the points were defined was omitted. Here the details of obtaining the interpolated values and the accuracy of the results will be given for all of the interpolation schemes used.

A.4.1 Formulations for Polynomial Interpolation

Here the interpolating polynomials for all cases up to cubic that are used or interpolation of model quantities at grid interfaces will be generated using the classical Lagrange formula,

$$P_{(N-1)}(x) = \frac{(x-x_2)(x-x_3)\dots(x-x_N)}{(x_1-x_2)(x_1-x_3)\dots(x_1-x_N)}u_1 + \frac{(x-x_1)(x-x_3)\dots(x-x_N)}{(x_2-x_1)(x_2-x_3)\dots(x_2-x_N)}u_2 + \dots + \frac{(x-x_1)(x-x_2)\dots(x-x_{N-1})}{(x_N-x_1)(x_2-x_3)\dots(x_N-x_{N-1})}u_N, \tag{A.104}$$

which gives a polynomial interpolation of degree $N-1$.

Since the ratio of resolutions useful — and therefore allowed — is restricted to 3:1, only the interpolating formulas specific to that case are necessary. There will be ten of them: four linear, four quadratic, and two cubic. Figure A.3 defines the quantities used in all of the cases.

The interpolating polynomials for the values u' and u'' at x' and x'', respectively, are needed for each of the possible combinations of the x_i that can occur in the problem, given the values u_i at those points.

The quantities necessary for the linear cases are $u'_{(1)}(u_1, u_2)$, $u'_{(1)}(u_2, u_3)$, $u''_{(1)}(u_2, u_3)$, and $u''_{(1)}(u_3, u_4)$. Using (A.104), these are given by

$$\begin{aligned} u'_{(1)}(u_1, u_2) &= \frac{(x' - x_2)}{(x_1 - x_2)}u_1 + \frac{(x' - x_1)}{(x_2 - x_1)}u_2 \\ &= \frac{(4-3)}{(0-3)}u_1 + \frac{(4-0)}{(3-0)}u_2 \\ &= -\frac{1}{3}u_1 + \frac{4}{3}u_2 \end{aligned} \tag{A.105}$$

$$\begin{aligned} u'_{(1)}(u_2, u_3) &= \frac{(x' - x_3)}{(x_2 - x_3)}u_2 + \frac{(x' - x_2)}{(x_3 - x_2)}u_3 \\ &= \frac{(4-6)}{(3-6)}u_2 + \frac{(4-3)}{(6-3)}u_3 \\ &= \frac{2}{3}u_2 + \frac{1}{3}u_3, \end{aligned} \tag{A.106}$$

and, by symmetry,

$$u''_{(1)}(u_2, u_3) = \frac{1}{3}u_2 + \frac{2}{3}u_3 \tag{A.107}$$

$$u''_{(1)}(u_3, u_4) = \frac{4}{3}u_3 - \frac{1}{3}u_4. \tag{A.108}$$

For the quadratic cases, the quantities necessary for the problem are $u'_{(2)}(u_1, u_2, u_3)$,

$u''_{(2)}(u_1,u_2,u_3)$, $u'_{(2)}(u_2,u_3,u_4)$, and $u''_{(2)}(u_2,u_3,u_4)$. Calculating these as above yields

$$\begin{aligned} u'_{(2)}(u_1,u_2,u_3) &= \frac{(x'-x_2)(x'-x_3)}{(x_1-x_2)(x_1-x_3)}u_1 + \frac{(x'-x_1)(x'-x_3)}{(x_2-x_1)(x_2-x_3)}u_2 \\ &\quad + \frac{(x'-x_1)(x'-x_2)}{(x_3-x_1)(x_3-x_2)}u_3 \\ &= \frac{(4-3)(4-6)}{(0-3)(0-6)}u_1 + \frac{(4-0)(4-6)}{(3-0)(3-6)}u_2 + \frac{(4-0)(4-3)}{(6-0)(6-3)}u_3 \\ &= -\frac{1}{9}u_1 + \frac{8}{9}u_2 + \frac{2}{9}u_3 \end{aligned} \tag{A.109}$$

$$\begin{aligned} u''_{(2)}(u_1,u_2,u_3) &= \frac{(x''-x_2)(x''-x_3)}{(x_1-x_2)(x_1-x_3)}u_1 + \frac{(x''-x_1)(x''-x_3)}{(x_2-x_1)(x_2-x_3)}u_2 \\ &\quad + \frac{(x''-x_1)(x''-x_2)}{(x_3-x_1)(x_3-x_2)}u_3 \\ &= \frac{(5-3)(5-6)}{(0-3)(0-6)}u_1 + \frac{(5-0)(5-6)}{(3-0)(3-6)}u_2 + \frac{(5-0)(5-3)}{(6-0)(6-3)}u_3 \\ &= -\frac{1}{9}u_1 + \frac{5}{9}u_2 + \frac{5}{9}u_3. \end{aligned} \tag{A.110}$$

And thus by symmetry,

$$u'_{(2)}(u_2,u_3,u_4) = \frac{5}{9}u_2 + \frac{5}{9}u_3 - \frac{1}{9}u_4 \tag{A.111}$$

$$u''_{(2)}(u_2,u_3,u_4) = \frac{2}{9}u_2 + \frac{8}{9}u_3 - \frac{1}{9}u_4. \tag{A.112}$$

There is only one combination of the points x_i for the cubic case, and that is the one using all four points. Values at both x' and x'' for the cubic case are needed, giving

$$\begin{aligned} u'_{(3)}(u_1,u_2,u_3,u_4) &= \frac{(x'-x_2)(x'-x_3)(x'-x_4)}{(x_1-x_2)(x_1-x_3)(x_1-x_4)}u_1 + \frac{(x'-x_1)(x'-x_3)(x'-x_4)}{(x_2-x_1)(x_2-x_3)(x_2-x_4)}u_2 \\ &\quad + \frac{(x'-x_1)(x'-x_2)(x'-x_4)}{(x_3-x_1)(x_3-x_2)(x_3-x_4)}u_3 + \frac{(x'-x_1)(x'-x_2)(x'-x_3)}{(x_4-x_1)(x_4-x_2)(x_4-x_3)}u_4 \\ &= \frac{(4-3)(4-6)(4-9)}{(0-3)(0-6)(0-9)}u_1 + \frac{(4-0)(4-6)(4-9)}{(3-0)(3-6)(3-9)}u_2 \\ &\quad + \frac{(4-0)(4-3)(4-9)}{(6-0)(6-3)(6-9)}u_3 + \frac{(4-0)(4-3)(4-6)}{(9-0)(9-3)(9-6)}u_4 \\ &= -\frac{5}{81}u_1 + \frac{60}{81}u_2 + \frac{30}{81}u_3 - \frac{4}{81}u_4. \end{aligned} \tag{A.113}$$

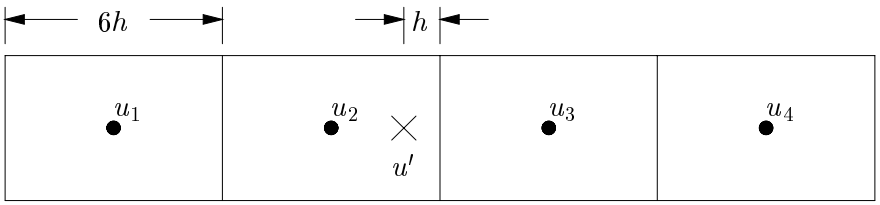

Figure A.4: Interpolation points

Then lastly, and also by symmetry,

$$u''_{(3)}(u_1, u_2, u_3, u_4) = -\frac{4}{81}u_1 + \frac{30}{81}u_2 + \frac{60}{81}u_3 - \frac{5}{81}u_4. \tag{A.114}$$

A.4.2 Accuracy of Polynomial Interpolation

A generalized form for the Taylor expansions of the known points, u_i, in Figure A.4 about the interpolation point, u', are used to write

$$u(x + nh, t) = u' + nhu'_x + \frac{n^2h^2}{2}u'_{xx} + \frac{n^3h^3}{6}u'_{xxx} + O(h^4) \tag{A.115}$$

$$u(x - nh, t) = u' - nhu'_x + \frac{n^2h^2}{2}u'_{xx} - \frac{n^3h^3}{6}u'_{xxx} + O(h^4). \tag{A.116}$$

Using no interpolating formula for u', but setting it to the nearest point, in this case u_2 gives

$$\begin{aligned} u'_{(0)} &= u_2 = u' - 2hu'_x + \frac{4h^2}{2}u'_{xx} - \frac{8h^3}{6}u'_{xxx} + O(h^4) \\ &= u' + O(h), \end{aligned} \tag{A.117}$$

which is a first-order accurate value for u'. Again substituting (A.115) and (A.116) for the terms in (A.106) gives

$$\begin{aligned} u'_{(1)}(u_2, u_3) &= \frac{2}{3}u'_2 + \frac{1}{3}u'_3 \\ &= \frac{2}{3}\left(u' - 2hu'_x + \frac{4h^2}{2}u'_{xx}\right) \\ &\quad + \frac{1}{3}\left(u' + 4hu'_x + \frac{16h^2}{2}u'_{xx}\right) + O(h^3) \\ &= u' + 4h^2u'_{xx} + O(h^3) = u' + O(h^2). \end{aligned} \tag{A.118}$$

And likewise expanding (A.105) gives

$$\begin{aligned} u'_{(1)}(u_1, u_2) &= -\frac{1}{3}u'_1 + \frac{4}{3}u'_2 \\ &= -\frac{1}{3}\left(u' - 8hu'_x + \frac{64h^2}{2}u'_{xx}\right) \\ &\quad + \frac{4}{3}\left(u' - 2hu'_x + \frac{4h^2}{2}u'_{xx}\right) + O(h^3) \\ &= u' - 8h^2u'_{xx} + O(h^3) = u' + O(h^2), \end{aligned} \tag{A.119}$$

showing second-order accuracy for linear interpolation. Moving on to (A.109),

$$\begin{aligned} u'_{(2)}(u_1, u_2, u_3) &= -\frac{1}{9}u_1 + \frac{8}{9}u_2 + \frac{2}{9}u_3 \\ &= -\frac{1}{9}\left(u' - 8hu'_x + \frac{64h^2}{2}u'_{xx} - \frac{512h^3}{6}u'_{xxx}\right) \\ &\quad - \frac{8}{9}\left(u' - 2hu'_x + \frac{4h^2}{2}u'_{xx} - \frac{8h^3}{6}u'_{xxx}\right) \\ &\quad - \frac{2}{9}\left(u' + 4hu'_x + \frac{16h^2}{2}u'_{xx} - \frac{64h^3}{6}u'_{xxx}\right) + O(h^4) \\ &= u' + \frac{32}{3}h^3u'_{xxx} + O(h^4) = u' + O(h^3) \end{aligned} \tag{A.120}$$

shows third-order accuracy for quadratic interpolation. And finally, expanding (A.113)

$$\begin{aligned}
u'_{(3)}(u_1,u_2,u_3,u_4) &= -\frac{5}{81}u_1+\frac{60}{81}u_2+\frac{30}{81}u_3-\frac{4}{81}u_4 \\
&= -\frac{5}{81}\left(u'-8hu'_x+\frac{64h^2}{2}u'_{xx}-\frac{512h^3}{6}u'_{xxx}+\frac{4096h^4}{24}u'_{xxxx}\right) \\
&\quad +\frac{60}{81}\left(u'-2hu'_x+\frac{4h^2}{2}u'_{xx}-\frac{8h^3}{6}u'_{xxx}+\frac{16h^4}{24}u'_{xxxx}\right) \\
&\quad +\frac{30}{81}\left(u'+4hu'_x+\frac{16h^2}{2}u'_{xx}-\frac{64h^3}{6}u'_{xxx}+\frac{256h^4}{24}u'_{xxxx}\right) \\
&\quad -\frac{4}{81}\left(u'+4hu'_x+\frac{16h^2}{2}u'_{xx}-\frac{64h^3}{6}u'_{xxx}+\frac{10000h^4}{24}u'_{xxxx}\right) \\
&\quad +O(h^5) \\
&= u'-\frac{80}{3}h^4u'_{xxxx}+O(h^5)=u'+O(h^4),
\end{aligned} \tag{A.121}$$

gives fourth-order accuracy for the cubic interpolation.

A.4.3 Cubic Spline Formulation

Cubic spline interpolation gives an interpolation formula which is smooth in the first derivative and continuous in the second derivative within a closed interval of interpolation. Given a set of known values, u_j, for $j=1,\ldots,n$, specified at the points x_j, assume that the second derivatives, u''_j, are also known. Then an interpolation formula can be written as

$$\tilde{u}(x)=Au_j+Bu_{j+1}+Cu''_j+Du''_{j+1}, \tag{A.122}$$

where primes indicate derivatives with respect to x, and the coefficients A, B, C, and D are defined as

$$A = \frac{x_{j+1} - x}{x_{j+1} - x_j} \tag{A.123}$$

$$B = 1 - A \tag{A.124}$$

$$C = \frac{A\left(A^2 - 1\right)\left(x_{j+1} - x_j\right)^2}{6} \tag{A.125}$$

$$D = \frac{B\left(B^2 - 1\right)\left(x_{j+1} - x_j\right)^2}{6}. \tag{A.126}$$

Notice that A and B are simply the coefficients of the linear Lagrange interpolating formula. Taking the first derivative of (A.122) gives

$$\tilde{u}' = \frac{u_{j+1} - u_j}{x_{j+1} - x_j} - \frac{\left(3A^2 - 1\right)\left(x_{j+1} - x_j\right)}{6} u_j'' + \frac{\left(3B^2 - 1\right)\left(x_{j+1} - x_j\right)}{6} u_{j+1}''. \tag{A.127}$$

Differentiating again gives

$$\tilde{u}'' = A u_j'' + B u_{j+1}'', \tag{A.128}$$

which shows that the second derivatives take their proper values at the known points, with a linear relationship in the intervals. However, the values of the second derivatives are not yet known. Using (A.127) in the intervals $[x_{j-1}, x_j]$ and $[x_j, x_{j+1}]$, set the first derivatives equal to each other at x_j. This step will not only provide a means for finding the values of u'', but guarantees the continuity of the first derivative in the interpolation formula. For the points $j = 2, \ldots, n-1$, the procedure yields

$$\frac{1}{6}\left[\left(x_{j+1} - x_j\right) u_{j+1}'' + 2\left(x_{j+1} - x_{j-1}\right) u_j'' + \left(x_j - x_{j-1}\right) u_{j-1}''\right] = \frac{u_{j+1} - u_j}{x_{j+1} - x_j} - \frac{u_j - u_{j-1}}{x_j - x_{j+1}}. \tag{A.129}$$

Or, if the grid spacing is constant, i.e. if $x_{j+1} - x_j = h$ for all values of j, then this can be written simply as

$$u_{j+1}'' + 4u_j'' + u_{j-1}'' = \frac{6}{h^2}\left(u_{j+1} - 2u_j + u_{j-1}\right). \tag{A.130}$$

At x_1 and x_n, the second derivatives cannot be found with this formula. The most common method is to set the second derivatives to zero at the boundaries, giving the "natural" spline. With these values, (A.129) and (A.130) give a set of n linear equations for the n unknown second derivatives.

Solving this system, known as a tridiagonal system, is a relatively straightforward process. The system of equations can be written in matrix form as

$$\begin{pmatrix} b_1 & c_1 & & & & & \\ a_2 & b_2 & c_2 & & & & \\ & a_3 & . & . & & & \\ & & . & . & . & & \\ & & & . & . & c_{n-2} & \\ & & & & a_{n-1} & b_{n-1} & c_{n-1} \\ & & & & & a_n & b_n \end{pmatrix} \begin{pmatrix} u_1'' \\ u_2'' \\ . \\ . \\ . \\ u_{n-1}'' \\ u_n'' \end{pmatrix} = \begin{pmatrix} d_1 \\ d_2 \\ d_3 \\ . \\ . \\ d_{n-1} \\ d_n \end{pmatrix}. \tag{A.131}$$

Then the solution process is given by a two step algorithm.

Algorithm A.4.1. Tridiagonal System Solution

$\alpha_1 \Leftarrow b_1$
$\beta_1 \Leftarrow \frac{d_1}{\alpha_1}$
for $j = 2$ to n
  $\alpha_j \Leftarrow b_j - a_j \frac{c_{j-1}}{\alpha_{j-1}}$
  $\beta_j \Leftarrow \frac{d_j - a_j \beta_{j-1}}{\alpha_j}$
$u_n'' \Leftarrow \beta_n$
for $j = n - 1$ to 1
  $u_j'' \Leftarrow \beta_j - \frac{c_j}{\alpha_j} u_{j+1}''$

With the values u_j and u_j'' known at the points x_j, the interpolation formula given by (A.122) can be used to obtain interpolated values within the interval $[x_1, x_n]$.

A.4.4 Cubic Spline Accuracy

Suppose the interpolated value is calculated for a point midway between two points where the actual value is known. The interpolated value is given by the formula

$$\tilde{u}_{j+\frac{1}{2}} = \frac{u_j + u_{j+1}}{2} - \frac{h^2}{16}\left(z_j + z_{j+1}\right), \tag{A.132}$$

where $h = x_{j+1} - x_j$, the values u_j and u_{j+1} are given, and the values z_j and z_{j+1} are the calculated second derivatives from the procedure given above. Expanding the first term about the interpolation point gives

$$\frac{u_j + u_{j+1}}{2} = u + \frac{h^2}{8}u'' + \frac{h^4}{192}u^{(4)} + \mathrm{O}\left(h^6\right). \tag{A.133}$$

Differentiating twice yields

$$\frac{u_j'' + u_{j+1}''}{2} = u'' + \frac{h^2}{8}u^{(4)} + \mathrm{O}\left(h^4\right), \tag{A.134}$$

which can be substituted back into (A.133) to give

$$\frac{u_j + u_{j+1}}{2} = u + \frac{h^2}{16}\left(u_j'' + u_{j+1}''\right) - \frac{h^4}{96}u^{(4)} + \mathrm{O}\left(h^6\right). \tag{A.135}$$

Now proceeding to the second term of (A.132), expand the interpolation formula about x_j to give

$$\frac{\tilde{u}\left(x_j + \delta\right) + \tilde{u}\left(x_j - \delta\right)}{2} = u_j + \frac{\delta^2}{2}u_j'' + \frac{\delta^4}{24}u_j^{(4)} + \mathrm{O}\left(\delta^6\right), \tag{A.136}$$

where the fourth derivative is defined only for $0 < \delta < h$. Since the second derivative of the interpolating formula is piecewise linear, the fourth derivative is zero. Remembering that the interpolation formula returns the known values u_j for all j, take the limit

$$\lim_{\delta \to h^-} \frac{\tilde{u}\left(x_j + \delta\right) + \tilde{u}\left(x_j - \delta\right)}{2} = \frac{u_{j+1} + u_{j-1}}{2} = u_j + \frac{h^2}{2}z_j + \mathrm{O}\left(h^6\right). \tag{A.137}$$

A similar expansion for $u(x)$ gives

$$\frac{u_{j+1}+u_{j-1}}{2} = u_j + \frac{h^2}{2}u_j'' + \frac{h^4}{24}u_j^{(4)} + \mathrm{O}\left(h^6\right). \tag{A.138}$$

Subtracting (A.138) from (A.137) gives

$$z_j = u_j'' + \frac{h^2}{12}u_j^{(4)} + \mathrm{O}\left(h^4\right). \tag{A.139}$$

Finally, substituting (A.139) and (A.135) into (A.132) gives

$$\tilde{u}_{j+\frac{1}{2}} = u + \frac{h^2}{16}\left(u_j'' + u_{j+1}''\right) - \frac{h^4}{96}u^{(4)} - \frac{h^2}{16}\left(u_j'' + u_{j+1}'' + \frac{h^2}{12}u_j^{(4)}\right) + \mathrm{O}\left(h^6\right) \tag{A.140}$$

$$= u + \mathrm{O}\left(h^4\right). \tag{A.141}$$

Thus cubic spline interpolation is fourth order accurate.

A.5 Filtering

In the ocean model, the allowable timestep is increased by filtering short wavelengths from the solutions at high latitudes. The previous sections have shown that the decreasing longitudinal grid spacing at high latitudes results in more restriction on the timesteps that will remain stable with various finite difference discretizations. Here an example will be given of how removing short wavelengths can increase the timestep allowed by the linear stability condition.

Discretizing the diffusion equation,

$$\frac{\partial u}{\partial t} = a\frac{\partial^2 u}{\partial x^2}, \tag{A.142}$$

by forward-in-time and centered-in-space finite differences gives

$$\frac{u_m^{n+1}-u_m^n}{k} = a\frac{u_{m+1}^n - 2u_m^n + u_{m-1}^n}{h^2}, \tag{A.143}$$

where k is the timestep and h is the grid spacing. The stability condition for wavenumber component b is found to be

$$k \leq \frac{h^2}{2a \sin^2 \frac{bh}{2}}. \tag{A.144}$$

The smallest resolvable wavelength is $2h$, corresponding to a wavenumber of $\frac{\pi}{h}$, and giving the stability condition

$$k \leq \frac{h^2}{2a}. \tag{A.145}$$

But the stability condition is progressively less restrictive for longer wavelengths. Therefore, if we filter out (by Fourier filtering or an equivalent method) the smallest wavelengths, the maximum allowable stable timestep increases.

For the discrete model, filtering is applied only at the higher latitudes. The lowest latitudes at which to starting filtering are set by a variable for flexibility, and a reference latitude, ϕ_{ref}, is specified also, usually just smaller than the starting latitude for the filter. Then for each strip in the region thus defined, the model variables to be filtered are Fourier transformed decomposed in the following manner, outlined in [50]. For a variable q defined on a strip with l cells at latitude ϕ, define a critical wavenumber by

$$k = l \frac{\cos \phi_j^T}{\cos \phi_{\text{ref}}}. \tag{A.146}$$

Then Fourier coefficients are calculated by

$$A_0 = \frac{1}{l} \sum_{i=1}^{l} q_i \tag{A.147}$$

$$A_l = \frac{2}{l} \sum_{i=1}^{l} q_i \cos \frac{2\pi m i}{l} \qquad \text{for } m = 1, \ldots, \frac{k}{2} - 1 \tag{A.148}$$

$$A_{\frac{k}{2}} = \frac{1}{l} \sum_{i=1}^{l} q_i \cos(\pi i) \tag{A.149}$$

$$B_l = \frac{2}{l} \sum_{i=1}^{l} q_i \sin \frac{2\pi m i}{l} \qquad \text{for } m = 1, \ldots, \frac{k}{2} - 1 \tag{A.150}$$

and the filtered variable is reconstructed by the truncated series

$$\tilde{q}_i = A_0 + \sum_{m=1}^{\frac{k}{2}} A_m \cos\frac{2\pi m i}{l} + \sum_{m=1}^{\frac{k}{2}-1} B_m \sin\frac{2\pi m i}{l}. \tag{A.151}$$

Due to the no flux boundary condition of tracers and the zero boundary condition on velocities, only the cosine and sine components are needed, respectively. For strips that are unbroken by land and thus periodic, both Fourier components are needed. By this procedure, wavelengths smaller than those which are stable at the reference latitude are eliminated in the filtering region. See [67] for more details on filtering in the context of the shallow-water equations.

The filtering procedure can introduce oscillations due to the Gibbs effect. This can be lessened by adding a weight to the Fourier series to act as a window and produce a smoother result. Also the filtering procedure is greatly complicated on parallel distributed memory architectures. Each strip may lie within one or more subdomains, and some may require communication beyond the nearest neighbors. Combined with the limited region in which filtering is applied, load balancing of the model and scalability become difficult problems to solve. See [46] for more on these issues in an atmospheric model.

In the global runs of Chapter 5, filtering is applied for the standard, or base, case at latitudes of 60 degrees and higher. The tracers and baroclinic momentum are filtered at each timestep, with a reference latitude of 51 degrees for tracers and 54 degrees for the momentum. However, barotropic momentum and surface height are not filtered, thus limiting the barotropic timestep directly and the baroclinic timestep through limitations on the amount of subcycling allowed by stability. See Section 5.1 and Section 5.5 for reasons, details, and implications.

Bibliography

[1] T. L. Acker, L. E. Buja, J. M. Rosinski, and J. E. Truesdale. User's guide to NCAR CCM3. Technical note NCAR/TN-421+IA, NCAR, 1996.

[2] Akio Arakawa. Computational design for long-term numerical integration of the equations of fluid motion: Two-dimensional incompressible flow. Part I. *Journal of Computational Physics*, 1:119–143, 1966.

[3] John Bell, Marsha Berger, Jeff Saltzman, and Mike Welcome. Three-dimensional adaptive mesh refinement for hyperbolic conservation laws. *SIAM Journal on Scientific Computing*, 15:127–138, 1994.

[4] M. J. Berger and P. Colella. Local adaptive mesh refinement for shock hydrodynamics. *Journal of Computational Physics*, 82:64–84, May 1989.

[5] Marsha J. Berger. Stability of interfaces with mesh refinement. *Mathematics of Computation*, 45(172):301–318, October 1985.

[6] Marsha J. Berger. On conservation at grid interfaces. *SIAM Journal on Numerical Analysis*, 24:967–984, October 1987.

[7] Eric Blayo and Laurent Debreu. Adaptive mesh refinement for finite difference ocean models: First experiments. *Journal of Physical Oceanography*, 29:1239–1250, June 1999.

[8] W. Bourke. A multilevel spectral model. Part I: Formulation and hemispheric integrations. *Monthly Weather Review*, 102:687–701, 1974.

[9] G. L. Browning, J. J. Hack, and P. N. Swarztrauber. A comparison of three numerical methods for solving differential equations on the sphere. *Monthly Weather Review*, 111:1058–1075, May 1989.

[10] Frank Bryan. Parameter sensitivity of primitive equation ocean general circulation models. *Journal of Physical Oceanography*, 17:970–985, July 1987.

[11] Kirk Bryan. A numerical method for the study of the circulation of the world ocean. *Journal of Computational Physics*, 4:347–376, 1969.

[12] Kirk Bryan, Syukuro Manabe, and Ronald C. Pacanowski. A global ocean-atmosphere climate model. Part II. the oceanic circulation. *Journal of Physical Oceanography*, 5:30–46, January 1975.

[13] K. Caldeira and P. B. Duffy. Sensitivity of simulated CFC-11 distributions in a global ocean model to the treatment of salt rejected during sea-ice formation. *Geophysical Research Letters*, 25(7):1003–1006, April 1998.

[14] K. Caldeira, G. H. Rau, and P. B. Duffy. Predicted net efflux of radiocarbon from the ocean and increase in atmospheric radiocarbon content. *Geophysical Research Letters*, 25(20):3811–3814, October 1998.

[15] M. D. Cox. A baroclinic numerical model of the ocean: Preliminary results. In R. O. Reid, editor, *Numerical Models of Ocean Circulation*, pages 107–118. National Academy of Sciences, 1975.

[16] Gokhan Danabasoglu and James C. McWilliams. Sensitivity of the global ocean circulation to parameterizations of mesoscale tracer transports. *Journal of Climate*, 8:2967–2987, December 1995.

[17] Gokhan Danabasoglu, James C. McWilliams, and William G. Large. Approach to equilibrium in accelerated global oceanic models. *Journal of Climate*, 9:1092–1110, May 1996.

[18] Eric Deleersnijder and Jean-Michel Campin. On the computation of the barotropic mode of a free-surface world ocean model. *Annales Geophysicae*, 13:675–688, 1995.

[19] P. B. Duffy and K. Caldeira. Three-dimensional model calculation of ocean uptake of bomb C-14 and implications for the global budget of bomb C-14. *Global Biogeochemical Cycles*, 9(3):373–375, September 1995.

[20] P. B. Duffy and K. Caldeira. Sensitivity of simulated salinity in a three-dimensional ocean model to upper ocean transport of salt from sea-ice formation. *Geophysical Research Letters*, 24:1323–1326, June 1997.

[21] P. B. Duffy, K. Caldeira, J. Selvaggi, and M. I. Hoffert. Effects of subgrid-scale mixing parameterizations on simulated distributions of natural C-14, temperature, and salinity in a three-dimensional ocean general circulation model. *Journal of Physical Oceanography*, 27(4):498–523, April 1997.

[22] P. B. Duffy and K. G. Caldeira. Sensitivity of simulated salinities in a three dimensional ocean general circulation model to vertical mixing of destabilizing surface fluxes. *Climate Dynamics*, 15(2):81–88, February 1999.

[23] P. B. Duffy, D. E. Eliason, A. J. Bourgeois, and C. C. Covey. Simulation of bomb radiocarbon in two global ocean general circulation models. *Journal of Geophysical Research—Oceans*, 100(11):22545–22563, November 1995.

[24] P. B. Duffy, P. Eltgroth, A. J. Bourgeois, and K. Caldeira. Effect of improved subgrid scale transport of tracers on uptake of bomb radiocarbon in the GFDL ocean general circulation model. *Geophysical Research Letters*, 22(9):1065–1068, May 1995.

[25] P. G. Eltgroth, J. H. Bolstad, P. B. Duffy, A. A. Mirin, H. Wang, and M. F. Wehner. Coupled ocean/atmosphere modeling on high-performance computing systems. In *Proceedings of the Conference on Parallel Processing for Scientific Computing*, 1997.

[26] Alan D. Fox and Stephen J. Maskell. Two-way interactive nesting of primitive equation ocean models with topography. *Journal of Physical Oceanography*, 25:2977–2996, December 1995.

[27] W. Lawrence Gates and Christopher A. Riegel. A study of numerical errors in the integration of barotropic flow on a spherical grid. *Journal of Geophysical Research*, 67(2):773–784, February 1962.

[28] W. Lawrence Gates and Christopher A. Riegel. Comparative numerical integrations of simple atmospheric models on a spherical grid. *Tellus*, 15(4):406–423, 1963.

[29] Adrian E. Gill. *Atmosphere-Ocean Dynamics*. Academic Press, 1982.

[30] I. Ginis, A. Richardson, and L. M. Rothstein. Design of a multiply nested primitive equation ocean model. *Monthly Weather Review*, 126:1054–1079, April 1998.

[31] George J. Haltiner and Roger Terry Williams. *Numerical Prediction and Dynamic Meteorology*. John Wiley and Sons, 1980.

[32] S. Hellerman and M. Rosenstein. Normal monthly wind stress over the world ocean with error estimates. *Journal of Physical Oceanography*, 13:1093–1104, 1983.

[33] J. Leith Holloway, Jr., Michael J. Spelman, and Syukuro Manabe. Latitude-longitude grid suitable for numerical time integration of a global atmospheric model. *Monthly Weather Review*, 101:69–78, January 1973.

[34] J. T. Houghton, G. J. Jenkins, and J. J. Ephraums, editors. *Climate Change: The IPCC Scientific Assessment*. Cambridge University Press, 1990.

[35] M. Iskandarani, D.B. Haidvogel, and J.P. Boyd. A staggered spectral element model with applications to the oceanic shallow water equations. *International Journal for Numerical Methods in Fluids*, 20:393–414, 1995.

[36] Peter D. Killworth. Topographic instabilities in level model OGCMs. unpublished.

[37] Peter D. Killworth, David Stainforth, David J. Webb, and Stephen M. Paterson. The development of a free-surface Bryan-Cox-Semtner ocean model. *Journal of Physical Oceanography*, 21:1333–1348, September 1991.

[38] H. L. Kuo and Jack Nordo. Integration of four-level prognostic equations over the hemisphere. *Tellus*, 11(4):412–424, 1959.

[39] Yoshio Kurihara. Numerical integration of the primitive equations on a spherical grid. *Monthly Weather Review*, 93(7):399–415, July 1965.

[40] Yoshio Kurihara, Gregory J. Tripoli, and Morris A. Bender. Design of a movable nested-mesh primitive equation model. *Monthly Weather Review*, 107:239–249, March 1979.

[41] Marc Laugier, Philippe Angot, and Laurent Mortier. Nested grid methods for an ocean model: A comparative study. *International Journal for Numerical Methods in Fluids*, 23:1163–1195, December 1996.

[42] S. Levitus. *Climatological Atlas of the World Ocean*. U.S. Government Printing Office, Washington, DC, 1982. NOAA Prof. Paper 13.

[43] Dan Martin and Keith Cartwright. Solving Poisson's equation using adaptivive mesh refinement. unpublished.

[44] James C. McWilliams. Modeling the oceanic general circulation. In *Annual Review of Fluid Mechanics*, volume 28, pages 215–248. 1996.

[45] A. A. Mirin, J. J. Ambrosiano, J. H. Bolstad, A. J. Bourgeois, J. C. Brown, B. Chan, W. P. Dannevik, P. B. Duffy, P. G. Eltgroth, C. Matarazzo, and M. F. Wehner. Climate system modeling using a domain and task decomposition message-passing approach. *Computer Physics Communications*, 84:278–296, 1994.

[46] A. A. Mirin, D. E. Shumaker, and M. F. Wehner. Efficient filtering techniques for finite-difference atmospheric general circulation models on parallel processors. *Parallel Computing*, 24:729–740, June 1998.

[47] Lie-Yauw Oey and Ping Chen. A nested-grid ocean model: With application to the simulation of meanders and eddies in the Norwegian coastal current. *Journal of Geophysical Research*, 97:20063–20086, December 1992.

[48] Joseph Oliger and Xiaolei Zhu. Stability and error estimation for component adaptive grid methods. *Applied Numerical Mathematics*, 20:407–426, 1996.

[49] Ronald C. Pacanowski. MOM 2 documentation, user's guide and reference manual. Ocean Technical Report 3, GFDL, 1995.

[50] Ronald C. Pacanowski and Stephen M. Griffies. MOM 3.0 manual. Draft version, ftp://ftp.gfdl.gov/pub/rcp/ALPHA, may 1999.

[51] Joseph Pedlosky. *Geophysical Fluid Dynamics*. Springer, 1979.

[52] Joseph Pedlosky. *Ocean Circulation Theory*. Springer, 1996.

[53] José Peixoto and Abraham H. Oort. *Physics of Climate*. American Institute of Physics, 1992.

[54] A. L. Perkins, L.F. Smedstad, D. W. Blake, G. W. Heburn, and A. J. Wallcraft. A new nested boundary condition for a primitive equation ocean model. *Journal of Geophysical Research*, 102(C2):3483–3500, February 1997.

[55] William H. Press, Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery. *Numerical Recipes in C: the Art of Scientific Programming*. Cambridge University Press, 1992.

[56] M. Rancic, R. J. Purser, and F. Mesinger. A global shallow-water model using an expanded spherical cube: Gnomonic versus conformal coordinates. *Quarterly Journal of the Royal Meteorological Society*, 122:959–982, 1996.

[57] Philip J. Rasch. Conservative shape-preserving two-dimensional transport on a spherical reduced grid. *Monthly Weather Review*, 122:1337–1350, June 1994.

[58] Patrick J. Roache. *Computational Fluid Dynamics*. Hermosa Publishers, 1972.

[59] Alber J. Semtner. Modeling ocean circulation. *Science*, 269:1379–1385, September 1995.

[60] Albert J. Semtner, Jr. and Robert M. Chervin. Ocean general circulation from a global eddy-resolving model. *Journal of Geophysical Research—Oceans*, 97:5493–5550, April 1992.

[61] G. D. Smith. *Numerical Solutions of Partial Differential Equations: Finite Difference Methods*. Clarendon Press, 1985.

[62] Joseph Peter Sobel. *Nested Grids in Numerical Weather Prediction and an Application to a Mesoscale Jet Streak*. PhD thesis, Pennsylvania State University, November 1976.

[63] Michael A. Spall and William R. Holland. A nested primitive equation model for oceanic applications. *Journal of Physical Oceanography*, 21:205–220, February 1991.

[64] G. Starius. On composite mesh difference methods for hyperbolic differential equations. *Numerical Mathematics*, 35:241–255, 1980.

[65] John C. Strikwerda. *Finite Difference Schemes and Partial Differential Equations*. Wadsworth and Brooks, 1989.

[66] Paul N. Swarztrauber, David L. Williamson, and John B. Drake. The cartesian method for solving partial differential equations in spherical geometry. *Dynamics of Atmospheres and Oceans*, 27:679–706, 1997.

[67] Lawrence L. Takacs and Ramesh C. Balgovind. High-latitude filtering in global grid-point models. *Monthly Weather Review*, 111:2005–2015, October 1983.

[68] K. Takano. A numerical simulation of the world ocean circulation: Preliminary results. In R. O. Reid, editor, *Numerical Models of Ocean Circulation*, pages 121–129. National Academy of Sciences, 1975.

[69] A. M. Treguier, J. K. Dukowicz, and K. Bryan. Properties of nonuniform grids used in ocean general circulation models. *Journal of Geophysical Research*, 101(9):20877–20881, 1996.

[70] K. Trenberth. *Climate System Modeling*. Cambridge University Press, 1992.

[71] William S. van Arx. *An Introduction to Physical Oceanography*. Addison-Wesley, 1962.

[72] George Veronis. Large scale ocean circulation. In *Advances in Applied Mechanics*, volume 13, pages 1–92. Academic Press, 1973.

[73] M. Wehner, A. Bourgeois, P. Eltgroth, P. Duffy, and W. Dannevik. Parallel coupled oceanic-atmospheric general circulation model. In *Proceedings of the 6th Workshop on Use of Parallel Processors in Meteorology*, 1994.

[74] M. F. Wehner, A. A. Mirin, P. G. Eltgroth, W. P. Dannevik, C. R. Mechoso, J. D. Farrara, and J. A. Spahr. Performance of a distributed memory finite difference atmospheric general circulation model. *Parallel Computing*, 21:1655–1675, 1995.

[75] Frank M. White. *Viscous Fluid Flow*. McGraw-Hill, 1991.

[76] Da-Lin Zhang, Hai-Ru Chang, Nelson L. Seaman, Thomas T. Warner, and J. Michael Fritsch. A two-way interactive nesting procedure with variable terrain resolution. *Monthly Weather Review*, 114:1330–1339, July 1986.

University of California
Lawrence Livermore National Laboratory
Technical Information Department
Livermore, CA 94551